2016

Knowledge-Empowered Probabilistic Graphical Models for Physical-Cyber-Social Systems

Pramod Anantharam
Wright State University

Knowledge-empowered Probabilistic Graphical Models for Physical-Cyber-Social Systems

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy

By

PRAMOD ANANTHARAM
B.S., Vishweshwaraiah Technological University, 2006

2016
Wright State University

WRIGHT STATE UNIVERSITY
GRADUATE SCHOOL

April 14, 2016

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Pramod Anantharam ENTITLED Knowledge-empowered Probabilistic Graphical Models for Physical-Cyber-Social Systems BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy.

Amit Sheth, Ph.D.
Dissertation Co-Director

Krishnaprasad Thirunarayan, Ph.D.
Dissertation Co-Director

Michael Raymer, Ph.D.
Director, Computer Science and Engineering Ph.D. Program

Robert E.W. Fyffe, Ph.D.
Vice President for Research and Dean of the Graduate School

Committee on
Final Examination

Shaojun Wang, Ph.D.

Payam Barnaghi, Ph.D.

Biplav Srivastava, Ph.D.

Cory Henson, Ph.D.

Shalini Forbis, MD, MPH

ABSTRACT

Anantharam, Pramod. Ph.D., Department of Computer Science and Engineering, Wright State University, 2016. Knowledge-empowered Probabilistic Graphical Models for Physical-Cyber-Social Systems.

There is a rapid intertwining of sensors and mobile devices into the fabric of our lives. This has resulted in unprecedented growth in the number of observations from the physical and social worlds reported in the cyber world. Sensing and computational components embedded in the physical world constitute a Cyber-Physical System (CPS). Current science of CPS is yet to effectively integrate citizen observations in CPS analysis. We demonstrate the role of citizen observations in CPS and propose a novel approach to perform a holistic analysis of machine and citizen sensor observations. Specifically, we demonstrate the complementary, corroborative, and timely aspects of citizen sensor observations compared to machine sensor observations in Physical-Cyber-Social (PCS) Systems.

Physical processes are inherently complex and embody uncertainties. They manifest as machine and citizen sensor observations in PCS Systems. We propose a generic framework to move from observations to decision-making and actions in PCS systems consisting of: (a) *PCS event extraction*, (b) *PCS event understanding*, and (c) *PCS action recommendation*. We demonstrate the role of Probabilistic Graphical Models (PGMs) as a unified framework to deal with uncertainty, complexity, and dynamism that help translate observations into actions. Data driven approaches alone are not guaranteed to be able to synthesize PGMs reflecting real-world dependencies accurately. To overcome this limitation, we propose to empower PGMs using the declarative domain knowledge. Specifically, we propose four techniques: (a) Automatic creation of massive training data for Conditional Random Fields (CRFs) using domain knowledge of entities used in *PCS event extraction*, (b) Bayesian Network structure refinement using causal knowledge from Concept Net used in *PCS*

event understanding, (c) knowledge-driven piecewise linear approximation of nonlinear time series dynamics using Linear Dynamical Systems (LDS) used in *PCS event understanding*, and (d) transforming knowledge of *goals* and *actions* into a Markov Decision Process (MDP) model used in *PCS action recommendation*.

We evaluate the benefits of the proposed techniques on real-world applications involving traffic analytics and Internet of Things (IoT).

Contents

List of Figures

List of Tables

ACKNOWLEDGMENTS

I'm greatly indebted to my advisor, Prof. Amit Sheth for encouraging me to start my PhD. I would not have pursued my PhD without Prof. Sheth's tremendous trust in my abilities. His trust in me lead to my playing a leading role in some projects and a great exposure to various aspects of interdisciplinary research. I feel privileged to have participated in discussions with Prof. Sheth on some cutting edge computational challenges and new computational frontiers to address these challenges. I'm extremely lucky to have such a great advisor who let me find and pursue my passion. I'm sincerely grateful to Prof. Sheth for providing me great opportunities to grow and flourish by creating an environment that fosters future thinkers and researchers.

I'm heavily indebted to my advisor, Prof. Krishnaprasad Thirunarayan (a.k.a Prof. T. K. Prasad) for hand-holding in initial years of my PhD when I needed the most. I have been very lucky to have him as one of my advisors and our every day informal chat over coffee gave me an opportunity to learn, defend, exchange, and follow-up new ideas related to my research topic. It's impossible for me to enumerate all the learning I had from my interactions with Prof. T. K. Prasad, but, a partial list includes curiosity, selflessness, strive for excellence, patience, professionalism, rigorousness, tenacity, and service to community by imparting knowledge to kids. I look at Prof. T. K. Prasad as an ideal scholar and would strive rest of my life to follow some of these principles.

I enjoyed every bit of my stay at the Ohio Center of Excellence in Knowledge-enabled Computing (Kno.e.sis) and had immense learning from my advisors and my colleagues. Special thanks to Cory Henson, my senior at Kno.e.sis, for being very welcoming when I joined my PhD program. Subsequently, working with him was an immense learning and rewarding experience. Thinking clearly and articulating complex ideas using simple but precise language are two of many things I learned from Cory. I'm also fortunate to have him as my mentor and as a colleague on my first job

after graduation. I must thank Dr. Biplav Srivastava, Dr. Payam Barnaghi, Dr. Shalini Forbis, and Dr. Shaojun Wang for being on my committee and providing me with valuable inputs.

I have been very fortunate to have met and worked with many of my colleagues throughout my PhD, contributing significantly to my growth. Special thanks to Dr. Tanvi Banerjee, Surendra Marupudi, Vaikunth Sridharan, Dan Vanuch, Harshal Patni, Vahid Taslimitehrani, Wenbo Wang, Lu Chen, Raghava Mutharaju, Pavan Kapanipathi, Ashutosh Jadhav, Hemant Purohit, Vinh Nguyen, Satya Sahoo, Pablo Mendis, Delroy Cameron, Meena Nagarajan, Ajith Ranabahu, Cartic Ramakrishnan, and Karthik Gomadam. I would like to thank the Kno.e.sis community and special thanks to Tonya Davis for tirelessly working for the well being of students.

This dissertation is based upon work supported by the National Science Foundation under Grant No. EAR 1520870 titled "Hazards SEES: Social and Physical Sensing Enabled Decision Support for Disaster Management and Response" and SoCS grant IIS-1111182 titled as "Social Media Enhanced Organizational Sensemaking in Emergency Response". Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. I would also like to acknowledge the EU FP7 Citypulse project contract number: 609035 for providing a broader motivation for this work.

DEDICATION

To my mother and father for their sacrifices, encouragement, and imparting values of life

To my wife for her great support and patience

1

Introduction

"There was 5 exabytes of information created between the dawn of civilization through 2003, but that much information is now created every 2 days, and the pace is increasing."

— Eric Schmidt, 2010

1.1 Growing Digital Universe

Increasing number of sensors and mobile devices are being connected to the Internet spanning a variety of domains such as City Management [155], Ambient Intelligence [36; 71], Fitness and Wellbeing [73], and System Health Monitoring [141; 14]. Figure 1.1(a) shows a progressive growth of connected devices on the Internet and this number is expected to reach 50 billion by 2020 [47]. This connection of devices, people, data, and services is referred to as Internet of Everything (IoE) [67]. IoE is projected to be a $19 trillion market by 2022 [67].

"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

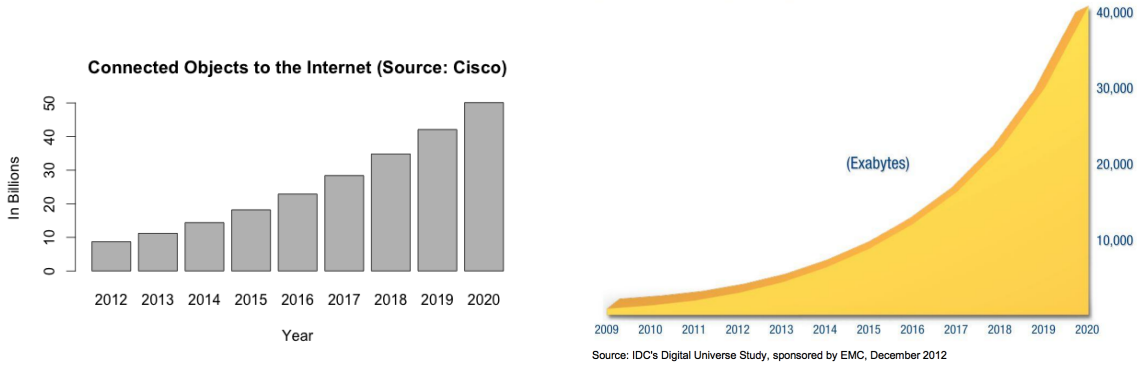
— Mark Weiser, 1991

Mark Weiser's vision of ubiquitous computing [148] in which computers are embedded deep into our life is increasingly being realized. Sensors and mobile devices with computational power are embedded in the physical world and they are increasingly indistinguishable from the fabric of our life. Currently, each vehicle has an average of 60-100 sensors on board. Because cars are rapidly getting "smarter" the number of sensors is projected to reach as many as 200 sensors per car [1]. The smart phones we carry has multiple sensors for recording location, movement, orientation, touch, and sound. Wearable devices can complement smart phones by observing sleep quality, activity level, heart rate, blood pressure, galvanic skin response, and respiration rate, resulting in unprecedented opportunities for *physical*, *physiological*, and *behavioral* understanding of people and their environment. Increasing availability and proliferation of sensors and mobile devices have resulted in massive amount of data being generated as shown in Figure 1.1(b). We need 1.5 billion 700 megabytes CD ROM disks to store 1 exabyte (10^{18} bytes) of data! Data generated by sensors and people include images, videos, numerical data, and textual data. Increasing availability of multimodal observations of the real-world has immense potential in enabling deeper understanding of the real-world and changes to it due to events.

What are real-world events? Throughout this work, we refer to the event as defined in the context of Physics[2]. Real-world events have a type, location, start-time, and end-time that can be represented as their meta data. These events exist in the physical world, i.e., we should be able to place a real-world event on a map (which is an abstraction of the physical space), and assign it a time point or a duration. For example, Super Bowl 2015 and Macy's Thanksgiving Day Parade 2015 are example of real-world events. Super Bowl 2015 is a *sporting match* (type) held at the University of Phoenix Stadium (location) on *February 1, 2015* (time). Macy's Thanksgiving Day Parade 2015 is a *parade* (type) start from *Central Park to Macy's Herald Square* (location), New York City starting at *9:00 a.m. Eastern Standard Time, Thanksgiving day* (time).

[1] http://www.automotivesensors2015.com/

[2] "A phenomenon or occurrence located at a single point in space-time, regarded as the fundamental observational entity in relativity theory." (from www.thefreedictionary.com/)



(a) Growth of number of devices connected to Internet over this decade reaching 50 billion by 2020

(b) Data growth from various connected devices and people in the digital universe

Figure 1.1: Digital Devices and Data Growth Projections (Source: http://bit.ly/1LgfMSb)

1.2 Real-world Events and its Multimodal Manifestations

Real-world events are observed by multiple observers including machine sensors and citizen sensors. For example, there may be sensors monitoring a road for number of vehicles passing over the road and people observing events in the city and reporting them on social media. We illustrate the characteristics of the observations related to real-world events by considering the domains of vehicular traffic, power grid maintenance, health and wellbeing, and system health monitoring.

1.2.1 Vehicular Traffic

A real-world event such as an *accident* on a road segment may manifest in observations made by various sources such as sensors monitoring the physical world (red box), official authorities reporting the incident (blue box), and people reporting the incident (green box). Figure 1.2 demonstrates an accident event type disrupting traffic at the intersection of *I-77 South* and *Ridgewood Road* on January 19, 2011. The impact of this real-world event on the movement of other vehicles on the road network is observed by sensors such as loop detectors that are closely monitoring the movement of vehicles in the *physical* world. People report disruptions in their mobility along with additional details in the *social* world as shown in the green box in Figure 1.2. Observations from

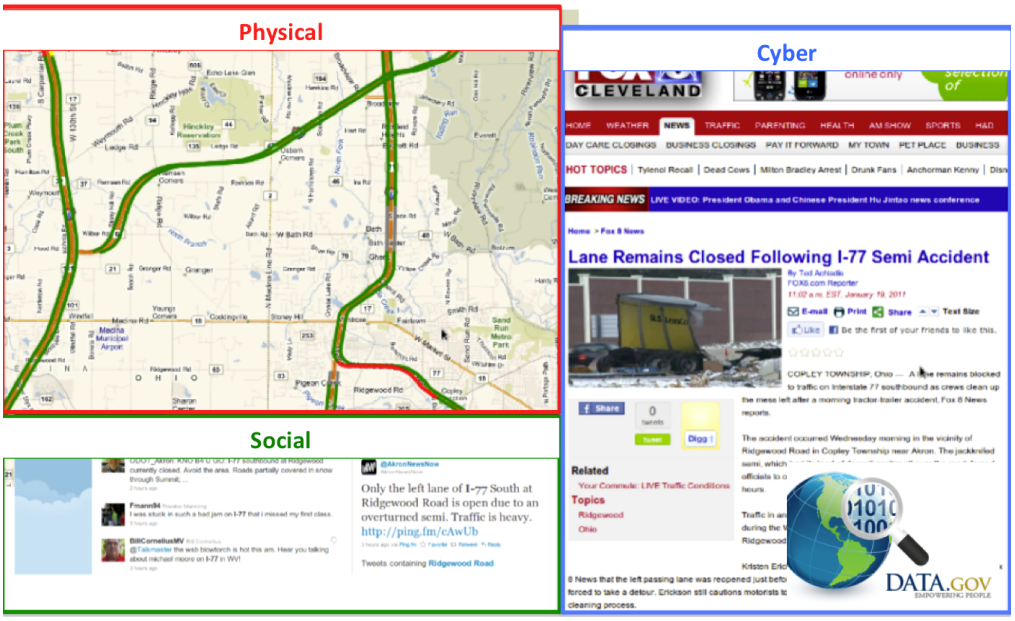


Figure 1.2: A real-world event of overturned semi (type: *accident*) at Ridgewood Rd. (location), on January 19, 2011 (time), reported by sensor, social, and a formal source

people (*" ... overturned semi ... "*) complement sensor data. When sensor data reports the same observation (slow moving traffic) in other situations such as sporting events, music events, or marathons, observations from people may be used to explain and/or disambiguate reduced speed observed in sensor data. Official report of the accident in the *cyber* world shown in the blue box of Figure 1.2 corroborates the observation from social data. In other words, we have heterogeneous observations related to the same event that needs to be combined and processed.

Another example of an event manifestation in the form of multimodal observations is shown in Figure 1.3 which involved data from 511.org and calendar of events published by the sporting arena at 7000 Coliseum Way, Oakland, CA 94621. We consider a road segment close to the sporting arena on *I-880 S* from street *66TH AVE* to *HEGENBERGER RD.* The average speed of vehicles passing through this link is reported to be 27 km/h as of *September 30th, 2012 at 2:20 PM* as shown in the Figure 1.3. This is considerably low compared to the speed limit of the link, which is 104 km/h,

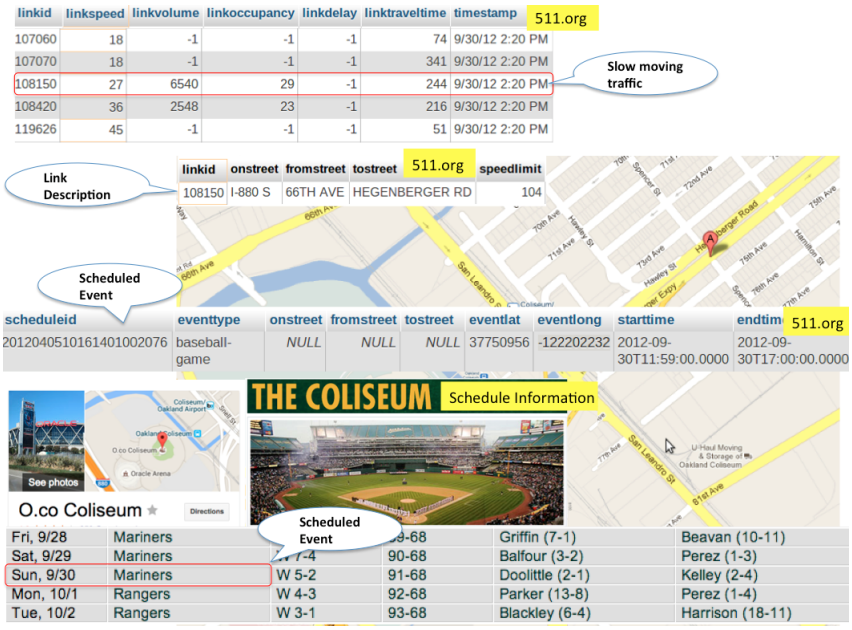


Figure 1.3: A real-world event from the domain of traffic, *sporting event* reported by sensor and the official calendar of the sporting arena (cyber) observations

indicating an anomaly. The calendar event published by the sporting arena provides an explanation for this reduced speed of vehicles passing through the link. Thus, the calender event of the baseball game of Seattle Mariners on *September 30th* provides a complementary information in understanding and explaining reduced average speed of vehicles reported by the sensors. Further, 511.org provides an account of important events in the city in the form of scheduled events. The *baseball game*[3] is provided as a scheduled event starting *September 30th, 12:00 PM* to *September 30th, 5:00 PM*, which complements the sensor data and corroborates the calendar event provided by the sporting arena.

(a) Report *before* flash over

(b) Report *during* flash over

(c) Report *after* flash over

(d) Effort to leverage social data for minimizing power outages (Photo via ShutterStock)

Figure 1.4: Power Grid Status Reported on Social Data

1.2.2 Power Grid Maintenance

On August 14, 2003, a massive blackout struck the Northeast United States resulting in loss of power for over 50 million people and $4 billion in lost revenue [144]. The problem was later traced to a contact between a sagging power line and a tree located in Cleveland. How can we prevent such blackouts? The role of social data in providing valuable and widely shared information on various aspects of power line safety is shown in Figure 1.4. In Figure 1.4(a), a person is reporting his observation of a tree touching a power line on twitter *before* any flash over of the power line. If a city authority acts on such public information, they can mobilize units to cut branches that are dangerously close to the power line avoiding impending power outages. Figure 1.4(b) shows a tweet reporting flash over of a power line due to a fallen tree. This information is valuable for city authorities to avoid further damage, and an immediate attention is required by city authorities. Figure 1.4(c) reports a destroyed power line due to a fallen tree along with precise location of the incident. This information would be valuable to city authorities when responding for quick restoration of power in a city. City power authorities can utilize such reports from social media data to complement their sensor-driven understanding of the power grid status in a geographical area. Figure 1.4(d) represents a call for preventive maintenance work due to volunteers reporting about locations of potential power grid hazard. There may be various sensors to monitor the loading and power flow in a power grid. These machine sensors report quantitative numbers and alarms can be configured when there are anomalous observation. However, city authorities need to leverage reports about the impact of real-world events on the power grid from social data, which may be complementary to sensor data. Consequently, they can be better positioned for dealing with preventive power grid maintenance or quick recovery for avoiding massive blackouts.

[3]Note that we just use event type to describe an event when the former unambiguously captures the event in the context of the discussion.

1.2.3 Health and Wellbeing

Prevalence of multimodal manifestations of health related events emphasize the role of integrating them in understanding and managing chronic health conditions such as asthma. Asthma is a multifaceted disease with complex interactions between the environment and the patient. Further, symptomatic variations across patients dictates a personalized and contextual understanding of asthma for each patient. Asthma is assessed on two orthogonal scales: (a) Asthma severity level and (b) Asthma control level [7]. While the patient does not change asthma severity level often, the control level of the patient may evolve dynamically in response to various personal and population level events. Asthma control level can take three possible states: (a) well-controlled, (b) moderately-controlled, and (c) poorly-controlled. Social data provides a great opportunity in tapping into health signals at both personal and population levels for a better understanding and management of asthma. Figure 1.5 demonstrates the role of social data in understanding environmental aspects and asthma related symptoms. Figure 1.5(a) reports a poor air quality incident along with a warning to people with asthma. Such alerts would be instrumental in providing timely information to patients with asthma for preventing asthma attacks. Figure 1.5(b) reports a personal level observation of a person who had an asthma attack resulting in disturbed sleep. Figure 1.5(d) is a similar report of asthma attack in the night by another person. Figure 1.5(c) reports an asthma attack possibly due to stress (which may be the cause of asthma attack). Such reports of asthma related information would complement sensor data and response to questionnaires [7] usually given to patients for assessing their health.

1.2.4 Use Cases: Observations

The three use cases presented in this section has the following qualities: i) *Heterogeneity*: All these domains exhibit heterogeneity in observations of a single event manifesting in multiple modalities requiring techniques to integrate and process heterogeneous observations. ii) *Qualitative vs. Quantitative*: Sensor data is qualitative providing precise observation of a quantity of interest, e.g., *27*

(a) Report of poor airquality

(b) Report of symptoms related to asthma and its impact on sleep

(c) Report of asthma attack due to possible stress

(d) Report of a severe asthma attack

Figure 1.5: Reports of Asthma Related Information on Social Data

km/h. Data from people are usually qualitative providing a high level description of an event, e.g., *overturned semi*. iii) *Explanation vs. Corroboration*: Data from people may explain sensor data, e.g., *overturned semi* explains the lower average speed of vehicles reported by sensors. If people are reporting *slow moving traffic*, then this information corroborates average speed of vehicles (*27 km/h*) reported by sensor data.

1.3 Physical-Cyber-Social (PCS) Systems

Most real-world events exhibit close interactions between physical, cyber, and the social worlds as illustrated by traffic analytics, power grid maintenance, and health and wellbeing applications. The nature of the physical world can be hard to understand by merely observing the physical world piecemeal e.g., space-time curvature was not obvious just by observing the planetary positions; only when Isaac Newton's theory failed to explain observed discrepancies (Mercury's orbit), that lead to the Theory of Relativity by Albert Einstein, which in-turn was a better theory to explain the physical world. Further, within our physical world, where Newtonian Physics is applicable, we may not have direct access to the physical world in most of the situations e.g., historically, astronomers observed celestial bodies and its relative movement to Earth for validating some theories since we don't have a global view of Earth and other celestial bodies. Understanding real-world events in the

physical world utilizing observational data is a challenging task. We highlight some of the challenges in understanding real-world events utilizing observational data.

Models in science play a central role for a scientific study and understanding of our physical world. Models in science are broadly classified as models of physical process and models of data. Along these lines, there are essentially two ways of understanding the physical world: i) Precise modeling of physical processes to describe the physical world in the form of mathematical models. For example, astronomers study the evolution of the universe by precisely modeling the movement of gas particles to simulate the evolution over billion of years. ii) The models of data allows us to infer theories of physical world through observations. For example, planetary orbits were observed first before any theory was proposed to explain their behavior. This thesis is dedicated to the second approach, where, we try to infer the models of the physical world from observational data. We strongly believe that precise mathematical models needs to be complemented with data from the real-world for a realistic understanding of the physical world.

> *"As far as the laws of mathematics refer to reality, they are not certain, as far as they are certain, they do not refer to reality."*
>
> — Albert Einstein, 1921

This quote by Albert Einstein reinforces our belief of utilizing observational data to deal with various complexities in the real-world which may not be precisely captured in mathematical equations. The real-world exhibits uncertainty, incompleteness, heterogeneity, and dynamism as some of the challenges described below.

1.3.1 Uncertainty

Uncertainty in the real-world are introduced at two levels: i) Stochasticity in the real-world: Real-world events may be inherently stochastic in nature, e.g., occurrence of accidents at various locations and time in a city may be stochastic. Stochasticity is partly due to the various factors such as weather

conditions (poor visibility, icy road), time of day, choices people make while driving, and state of alertness of the drivers. At this level of complexity, it would be impossible to collect all possible factors causing accidents resulting in a stochastic system. ii) Observational uncertainty: Observers of the real-world events are machine sensors and people. People reports may be biased, unreliable, and subjective. Machine sensors may be faulty or compromised.

1.3.2 Incompleteness

Incompleteness in observations introduce challenges in understanding the events in the physical world. Incompleteness can be broadly attributed to three sources: i) Incompleteness due to domain: Some real-world events may not be observed directly such as occurrence of a disease. Only symptoms are observed which can then be used to conduct further tests resulting in additional observations utilized to infer the disease. Similarly, in the domain of traffic analytics, we may observe slow moving traffic without access to the real-world event (e.g., icy road) that may have caused the slow moving traffic. ii) Incompleteness due to coverage: The disappearance of Malaysia Airlines flight MH370 on 8th of March 2014 is a real-world example of incomplete information [17] that baffled the search teams. The missing information is primarily due to limited coverage of flight tracking technologies. iii) Incompleteness due to failure: The Aeroperú Flight 603 disaster on 2nd October 1996 is an example of sensor (faulty altimeter) failure resulting in incomplete information to the pilots [149] leading to a disastrous outcome.

1.3.3 Heterogeneity

Tight intertwining of physical, cyber, and social worlds results in multimodal observations of the real-world events. For example, an accident event in the real-world may manifest as slow moving traffic in sensors monitoring average speed of vehicles passing through that link, a sequence of images in the traffic camera, formal incident reports from law enforcement, news report, and textual report of accident on social data. Machine sensors report numerical observations and people report textual

observations. Further, there may be a variety of machine sensors for monitoring various aspects of the physical world. All these observations result in heterogeneity in the observations reported for real-world events.

1.3.4 Dynamism

Real-world systems such as road network in a city are subject to rapid changes due to various stimulus such as occurrence of accident, changing road conditions, time of day, day of week, and construction schedule. These rapid changes and interactions between the stimulus results in dynamic behavior and evolution of these systems. Understanding the impact of real-world events on the dynamics of the system is crucial for analyzing and predicting the behavior of the system. For example, if we could recognize an impending major traffic jam based on the dynamics, city authorities can take corrective actions. Similarly, in the context of asthma management, patient's control level is subject to changing dynamic stimulus such as varying physical activity, pollen level, air quality index, temperature, humidity, and medication intake. Understanding the control level dynamics is crucial for avoiding impending asthma attacks.

PCS computing encompasses *horizontal* and *vertical* operators to deal with heterogeneity and volume challenge of data being generated by PCS systems [130]. The *horizontal* operator performs a semantic integration of heterogeneous observations spanning physical, cyber, and social domains. E.g., in the domain of traffic analytics, a horizontal operator performs an integration of numerical data from sensors with textual data from people. The *vertical* operator abstracts massive raw data into succinct summaries of human intelligible abstractions e.g., associating meaning with numerical time series data (physiological observations in asthma management scenario) in the form of abstractions such as the control level (well-controlled, moderate-control, and poor-control) would lead to immediate action by doctors/patient. PCS computing refers to the computational techniques that combine multimodal observations from physical, cyber, and social worlds for a holistic understanding of real-world events. The social data (textual observations) is indispensable for obtaining

complementary and corroborative information to sensor data (numerical observations). Based on the challenges in processing observational data for understanding real-world events in PCS systems, we present the thesis contributions in the next section.

1.4 Processing Manifestations of Real-world Events

> *"Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering – uncertainty and complexity ..."*
>
> — Michael Jordan, UC Berkeley, 1998

Understanding real-world events from observational data is a challenging problem encompassing *uncertainty*, *incompleteness*, *heterogeneity*, and *dynamism*. In this dissertation, we demonstrate the natural fit of probabilistic graphical models (PGMs) in dealing with some of the PCS challenges. PGMs utilize probability as a calculus to deal with *uncertainty* and graphical structure to model interactions between real-world events. We present techniques to integrate multimodal observations such as sensor data (numerical) and social data (textual) to deal with the challenges of *incompleteness* and *heterogeneity*. We formulate time series based probabilistic models to capture *dynamism* of real-world events.

Understanding real-world event manifestations in domains such as traffic understanding, power grid maintenance, and health and wellbeing requires processing of massive amounts of heterogeneous observations. These observations contain valuable nuggets of information for decision makers such as city authorities, doctors, and patients. For example, if city authorities know the reason for slow moving traffic, they can mobilize appropriate units for corrective actions minimizing the impact on the mobility of people. Similarly, if a pediatrician knows the triggers leading to asthma exacerbations in a child, she can recommend corrective actions to minimize asthma attacks in future. William Pollard, a English writer stated the futility of having lot of information almost a century ago.

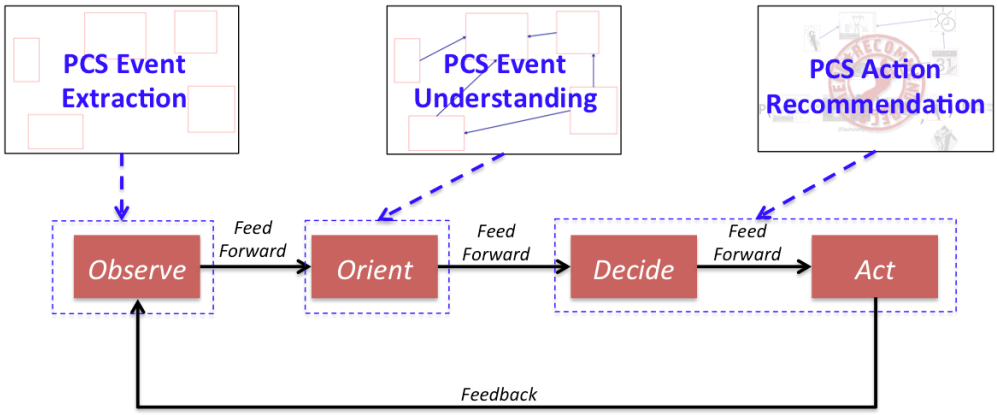


Figure 1.6: (Image credit: Wikipedia) The OODA loop proposed by Colonel John Boyd which is typically followed by an individual or an organization for making intelligent decisions

> *"Information is a source of learning. But unless it is organized, processed, and available to the right people in a format for decision making, it is a burden, not a benefit."*
>
> — William Pollard, (1828 – 1893)

A holistic approach to Analytics in PCS systems can allow decision makers to transform observations to actions. In a PCS system, we need to *observe* the physical environment (using multi-modal data), orient ourselves properly in the context of the physical environment (interpreting multi-modal data), *decide* (using a model of action for a particular situation), and *act* (through actuation or recommending action to a person) in the environment. This cycle of observation, orient, decide, and action is formalized in a cycle called the OODA (Observe, Orient, Decide, and Act) loop proposed by US military strategist, Colonel John Boyd [29]. This model was proposed for decision making in complex combat situations encountered by fighter pilots. According to Boyd, when a pilot performs the loop quickly and accurately, the pilot can win the battle with an adversary. Though this theory was proposed in the context of a military conflict, we use this loop subconsciously in many real-world situations, e.g., while playing tennis, we strategically place the ball in the court after we observe the position of the opponent. Figure 1.6 depicts various components of the OODA

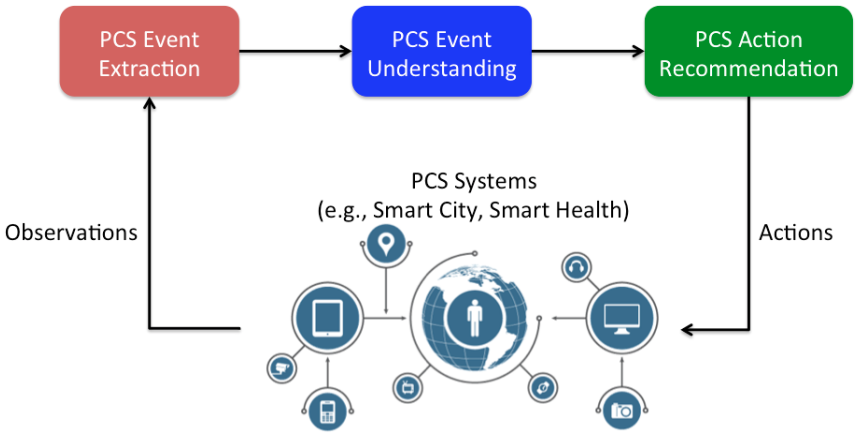


Figure 1.7: Processing observations from a PCS System in three steps: PCS Event Extraction, PCS Event Understanding, and PCS Action Recommendation

loop. There are feedback loops from orient, decide, and act phases demonstrating their iterative nature. The *observe* phase involves gathering data from machine sensors (numerical) and people (textual). In the *orient* phase, the collected data is interpreted based on the prior knowledge based on codified past experiences. The *decision* phase involves choosing the action based on the goal and current situation. The *act* phase involves carrying out remedial or proactive or preventive actions.

With conceptual underpinnings in the OODA loop, we present a three step approach to move from observations to actions in PCS Systems as shown in Figure 1.7. i) *PCS Event Extraction* involves defining events of interest in a domain and extracting events from observational data (both textual and numerical). This step requires techniques to process textual observations reported on PCS Systems. This step maps to the *observe* phase of OODA Loop. ii) *PCS Event Understanding* maps to *orient* phase of the OODA Loop. First, we need techniques to integrate textual observations and its manifestations in sensor data utilizing their co-existence in space and time. Next, we need techniques to infer possible interactions between various extracted events. iii) *PCS Action Recommendation* deals with navigating a task space to recommend best possible action toward a goal. This step maps to the *decide* and *act* phase of the OODA Loop.

The rest of the chapters in this dissertation are dedicated describing the three steps we outlined in Figure 1.7. To clarify the work presented in this dissertation, we outline the Why? What? and How? of this dissertation.

> *"People don't buy what you do; they buy why you do it. And what you do simply proves what you believe"*
>
> — Simon Sinek, Start with Why: How Great Leaders Inspire Everyone to Take Action

Why? addresses the purpose of the proposed work in a broader setting. What? provides details on the goals to be accomplished to deal with real-world events in PCS systems. How? provides a brief overview of the techniques we have developed in this dissertation to address the challenges offered by real-world events in PCS systems.

1.4.1 Why?

1. Understanding real-world event interactions and dynamics is crucial for informed decision making in many domains such as traffic analytics, power grid maintenance, healthcare, and system health monitoring.

2. Explaining real-world events requires a holistic analysis of observations spanning numerical and textual data due to the multimodal nature of real-world event manifestations. Further, numerical and textual data can be i) *complementary:* One modality source provides additional information compared to the other modality source, ii) *corroborative:* One modality source supports the other modality source, and iii) *timely:* One modality source may report an event before the other modality source.

3. Providing actionable information to decision-makers in a dynamic environment like PCS systems can be valuable for timely and informed decision making.

1.4.2 What?

We propose three research steps in processing observations from PCS systems and state some research questions for each step.

1. *Extracting events from its manifestations* (OODA loop equivalent: *observe*); What are the events of interest? How do they manifest in observational data? How can we infer events from observational data? What is the role of domain knowledge in event extraction?

2. *Understanding interactions between various events* (OODA loop equivalent: *orient*); How do events influence one-another? How do we infer the interactions from observational data across various modalities (numerical and textual data)? What is the role of domain knowledge in event understanding?

3. *Recommending actions based on understanding of events* (OODA loop equivalent: *decide* and *act*); How do we utilize our derived understanding? How can we recommend actions based on that understanding? What is the role of domain knowledge in action recommendation?

1.4.3 How?

1. For extracting events from its manifestations, we propose i) techniques to annotate short text messages using a sequence labeling model (Conditional Random Field), ii) algorithm to consume annotated short text messages with space-time information to extract real-world events, iii) techniques to leverage declarative domain knowledge of locations and event types in automated creation of large training data, and iv) techniques to extract events from SMS messages which is a popular mode of communication in developing countries.

2. For addressing the understanding of event interactions and their dynamics, we i) formalize the problem of understanding event interactions using Bayesian Network structure extraction from PCS system observations, ii) acknowledge the limitations of relying on data alone for

understanding event interactions and propose a hybrid approach to refine Bayesian Network structure utilizing declarative domain knowledge iii) formalize the problem of modeling non-linear time series dynamics utilizing a piece-wise linear segmentation of the time series modeled using a linear model (Linear Dynamical System) iv) propose algorithm to explain time series dynamics in sensor utilizing events extracted from short text messages reported by people.

3. For recommending best possible actions based on our domain understanding, we i) propose an algorithm that can consume a declarative domain specification of pre- and post-conditions of tasks and transform it to a sequential decision making problem (Markov Decision Process), ii) utilize the notion of best possible action from Markov Decision Process to recommend actions to the user.

1.5 Thesis Statement

Observations from diverse modalities can provide complementary, corroborative, and timely information about events in Physical-Cyber-Social systems. Probabilistic Graphical Models with the help of declarative domain knowledge provides an effective mechanism to: (a) uncover and interpret multimodal event manifestations in social and sensor data, (b) explore event interactions and dynamics, and (c) formalize optimal action recommendation in Physical-Cyber-Social systems.

1.6 Thesis contributions

We organize the contributions of this dissertation into technical contributions and broader impact.

1.6.1 Technical Contributions

We present specific details on the algorithms that were developed for PCS Systems utilizing PGMs and declarative domain knowledge. Technical contributions of this dissertation can be summarized

as follows:

1. We formalize PCS *event extraction*, *event understanding*, and *action recommendation* utilizing PGMs.

2. We integrate *sensor* (numerical) and *social* (textual) data for enhanced situational awareness of PCS systems using:

   (a) Algorithm to identify location and event terms in textual observations to generate annotated data

   (b) Algorithm to extract events from annotated data

   (c) Modeling sensor data dynamics utilizing probabilistic time series models

   (d) Characterize normalcy in sensor data dynamics and build personalized anomaly detection models

   (e) Algorithm to explain anomalies in sensor data dynamics utilizing events extracted from textual observations

3. We complement PGMs with declarative domain knowledge relevant to the PCS systems for:

   (a) *Event extraction*: Creation of a large training dataset utilizing declarative domain knowledge of traffic events and locations in a city to train Conditional Random Field (CRF) model for entity spotting.

   (b) *Event understanding*: Refining the structure of the Bayesian Network (BN) that represents the interactions between various real-world events utilizing the commonsense knowledge from ConceptNet and utilizing the knowledge of traffic domain for piecewise linear approximation of non-linear traffic dynamics using Linear Dynamical System (LDS).

   (c) *Action recommendation*: Parameterizing a Markov Decision Process (MDP) model utilizing declarative knowledge of tasks and their pre- and post-conditions in a domain.

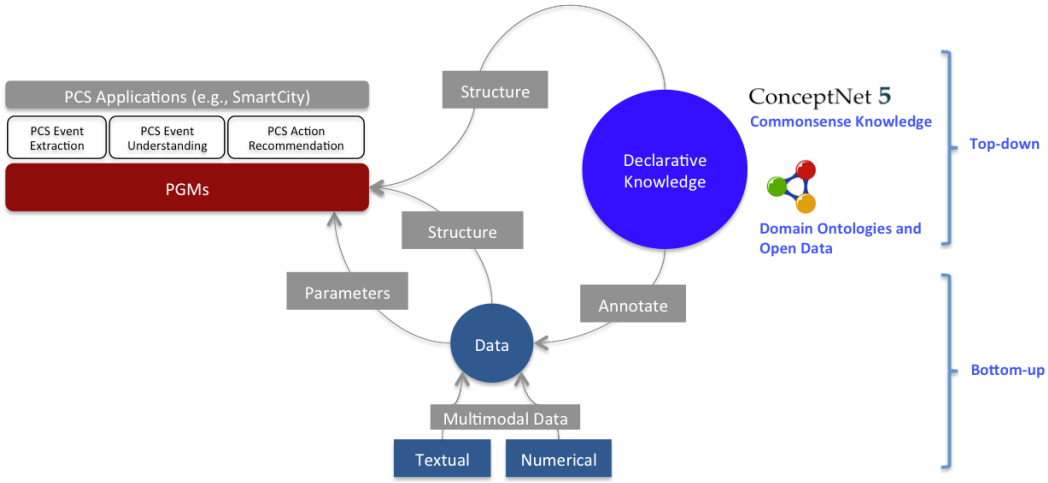


Figure 1.8: Top-down and bottom-up approach to building PGMs for PCS systems along with the multi-modal data of PCS system and Declarative Knowledge to empower PGMs.

1.6.2 Broader Contributions

This dissertation presents research on the role of textual observations from social data with respect to numerical observations from sensor data in PCS Systems. Throughout the dissertation, we utilize PGMs as a unifying representation for modeling real-world events and their dynamics. We highlight the role of declarative knowledge in complementing structure and parameter learning in PGMs.

> *"People often assume that all of the answers are in the data, and that domain knowledge is a secondary concern. In fact, it's often the other way around. The data provides the context to make a decision."*
>
> — Hilary Mason, Founder at Fast Forward Labs, 2013

Complementing PGMs with declarative domain knowledge is sought throughout the model building step of PCS Event Extraction, PCS Event Understanding, and PCS Action Recommendation. For clarity, Figure 1.8 presents the relationships between PGMs, declarative knowledge, and multimodal data in PCS Systems. Data is multimodal in PCS systems with both observations from

people (textual data) and from machine sensors (numerical data). *Structure* and *parameters* of a PGM may be learned from data alone (discussed in Chapter 3) but domain knowledge complements the structure extraction/specification process. Domain knowledge may also be utilized for parameter specification of PGMs. We demonstrate that PCS Event Extraction, PCS Event Understanding, and PCS Action Recommendation can be supported by utilizing a unified framework of knowledge-empowered PGMs.

In the rest of the dissertation, we present details on the three steps we presented in Figure 1.7. In Chapter 2, we present related work on CPS and summarize state-of-the-art in dealing with citizen observations in CSP. A background of PGMs is presented in Chapter 3 before we delve into PCS Event Extraction, PCS Event Understanding, and Action Recommendation in Chapters 4, 5, and 6 respectively, where, we explain the PGMs utilized and the role of declarative knowledge. Finally, we conclude and present future research direction in Chapter 7.

2

Related Work

We summarize the related research thrusts under three topics: (a) event extraction in Physical-Cyber-Social Systems (PCS), (b) event understanding in PCS, and (c) action recommendation in PCS.

2.1 Event Extraction in PCS

There are two orthogonal factors that influence the event extraction techniques: formal/informal text and open/closed domain nature of events. For both formal and informal textual data, we organize the related work into (a) open domain event extraction, where, the types of events are not known a priori and (b) closed domain event extraction, where, the types of events are known a priori.

Say, we want to extract all the events from a city related to the city infrastructure. There are two ways of specifying this problem: (a) Closed domain: Assume that we already know the events of interest e.g., events related to transportation network, power grid, water quality, etc. With this assumption, we need to device techniques to extract known events of interest from textual data. Known events can be in the form of a vocabulary or an ontology and extraction can be carried out using simple syntactic pattern matching or semantic extraction of events. Such an approach is

called closed domain. (b) Open domain: Assume that we don't know all the events of interest e.g., we cannot possibly list all the events related to power grid that may be of interest to the decision makers. With this assumption, we need to device techniques to identify events of interest and then extract them from textual data.

Further, the textual data from which we need to extract events in both the cases is another dimension. Textual data may include news articles and incident reports from formal sources in a city which, includes grammatical text. Such data sources–called formal text, may be easy to process for extracting events. However, the textual data may also be from people reported observations on social media such as twitter. Such data sources–called informal text, is messy due to its short length, short forms, redundancies, and informal language.

2.1.0.1 Formal Text

Event extraction from grammatical text such as news documents has been explored extensively. Parts of speech information and sentence parsing can be exploited in processing this type of content.
Open Domain: Event identification using a combination of text classification and use of named entities from news articles has been carried out by [79]. Similarly, to alleviate information overload in daily news, key entity and significant event extraction is done on news documents in [91]. A bipartite graph is induced based on the entities and their associations to documents using mutual reinforcement principle capturing salient entities and the documents with salient entities to rank the news events. Extraction of local events from blog entries has been carried out by [110].
Closed Domain: Use of known lightweight patterns to extract global crisis events from news text is presented in [139]. A combination of patterns specified manually and learned from data are utilized to determine event specific semantic roles (e.g., date and location, actors, event type). Known patterns for event specific roles are then used by event aggregation algorithms. An evaluation of accuracy of event extraction was carried out on a news corpus, and twenty seven out of twenty nine violent events in the test dataset were detected using the approach in [139]. Event extraction in the

context of detecting infectious disease outbreak was done by [56]. The event schema consisted of date range, geo-location, disease name, organism type and number affected by the disease, and the organism survival information. The event extraction is done by finite-state pattern matching the tokenized input text. The extracted events are compared against ground truth from ProMed[1] and WHO Infectious Disease Reports[2]. Creation of succinct summaries of events from news sources was carried out by [109]. A hierarchical clustering algorithm to cluster sentences referring to the same event has been presented as a baseline. Sentences in a news article that do not really describe the event are filtered out before clustering to obtain improved clustering accuracy in [109].

2.1.0.2 Informal Text

Event extraction from user-generated content with no overt structure which contains lot of slangs and non-standard abbreviations can be done using techniques that differ from those described in Section 2.1.0.1. These text fragments may not follow any rules of grammar making it hard to process using traditional techniques.

Open Domain: Event extraction from informal text such as tweets has received increased research attention recently. Synthesizing subgraphs in a graph of keywords (nodes representing keywords and edges representing co-occurrence statistics) using community detection techniques is studied by [126]. Each subgraph formed by a community of keywords can represent an event. Clustering based approach to detect events and adapting it to streaming data is presented in [2]. This clustering based approach caters to open domain event extraction where there is no prior knowledge on the number of event types. Event extraction techniques are organized based on four tasks in [42]: New event detection, event tracking, event summarization, and event association. New event detection techniques are used to identify first story of an event. Event tracking captures the evolution of an event. Event summarization involves creating summaries from bursts of messages. Event association uncovers relationships between events leading to domain insights. Open domain extraction of events from

[1] http://www.promedmail.org/

[2] http://www.who.int/topics/infectious_diseases/en/

informal text is addressed in [122]. This work demonstrates that building a calendar of significant events is feasible using twitter stream using an unsupervised approach to process tweets and extract event types such as sports, concert, protests, politics, TV, and religion. The approach models each entity in terms of a mixture of event types and each event type in terms of a mixture of entities. It requires minimal supervision for labeling the event descriptors but provides a fairly convincing approach to handle noisy, redundant, and informal nature of tweets. The evaluation compares it with a supervised baseline with improvement in both precision and recall. Using tweets for predicting hit and run crimes has been proposed by [146]. A latent topic based model is constructed over semantic role labels [97] of events from tweets. A generalized linear regression model learns the association between topics and crimes from a training dataset. The Receiver Operating Characteristics (ROC) curve based evaluation compares this approach with a baseline that associates uniform priors to crimes on all days.

Closed Domain: Using twitter streams to estimate the occurrence of events and its intensity using a supervised learning approach has been proposed by [83]. They use an optimized feature selection approach coupled with regression to estimate the intensity of events based on event markers. An evaluation based on ground truth from rain gauges is used for validation. They also extend the evaluation to identify Influenza Like Illness and compare it with the data from Health Protection Agency[3]. The study concludes the feasibility of using tweets for estimating events and its intensity. A clustering based approach is used by [18] for distinguishing tweets related to real-world events from non-event tweets. Temporal (volume changes), social (replies, broadcast), topical (coherence of clusters), and twitter-centric (multi-word hashtags) features explored via clustering are utilized to inform a classifier that performs better than the Naive Bayes classifier used as a baseline.

Although event extraction from social media streams has received significant attention, there is very little work on identifying various events which impacts traffic flow in a city. Most of the event extraction techniques presented as related work do not emphasize location and duration of the real-

[3]http://www.hpa.org.uk/

world events except for [83]. We believe this is crucial for understanding city traffic events. Knowing the location and duration of city traffic events, such as traffic jam, is important for both decision makers and citizens for informed decision making. Impact assessment of events provide insights into the extent to which events disrupt traffic and allows city authorities to prioritize resources. Efforts reported so far lack integrated use of event localization, event duration, and impact assessment. In this dissertation, we develop techniques to extract city traffic related events from twitter data by emphasizing location, duration, and impact of real-world city traffic events. Events in the physical world manifests in various modalities including machine (numerical) and social (textual) observations spanning physical, cyber, and social domains. Analyzing multimodal observations is not the focus of the state-of-the-art presented for event extraction. We propose algorithms to extract real-world events from spatio-temporal social data.

2.2 Event Understanding in PCS

Exploring event interactions and event dynamics is referred to as *event understanding* in the context of PCS systems. *Event interactions* in the real-world can provide insights into the workings of the physical world. Decision makers can intervene based on the knowledge of event interactions to achieve a desired effect. For example, if we uncover that fog results in increased accidents at a particular location, decision makers can post warnings around that location in an attempt to minimize accidents. Real-world events often evolve over time and the evolution manifests in multimodal observations. Understanding *event dynamics* in PCS systems includes deriving explanations to evolving event manifestations in sensor data. Specifically, we develop techniques to interpret average speed and travel time variations in sensor data utilizing traffic related events extracted from textual data.

2.2.1 Understanding Event Interactions

We summarize the related work on uncovering event interactions from observational data.

2.2.1.1 General Approaches

Bayesian Network structure extraction from data is utilized to uncover event interactions. Reliance on data alone for structure extraction may not always lead to faithful uncovering of event interactions in the real-world and may lead to ambiguity, e.g., two event interaction configurations may have the same score. Use of domain knowledge in the construction of Bayesian Networks has been studied toward overcoming reliance on data alone. Utilizing inequality and range constraints on the statistical variables in a domain for parameter learning is explored in [88]. These constraints, provided by domain experts, are incorporated in the learning process using a constrained EM learning algorithm which minimizes the violation of constraints. A systematic process of transforming an ontology into Bayesian network is presented for modeling oesophagus cancer [59]. Use of ontologies in construction of Object Oriented Bayesian Network (OOBN) [76] is studied by several research efforts [84; 66; 50; 65]. These approaches leverage object oriented techniques for representation and reasoning in modeling large and complex domains.

2.2.1.2 Traffic Analytics

The research on traffic flow analysis can be categorized into two broad categories. The first category deals with traffic flow analysis considering only the sensor observations monitoring the road network [60; 123]. They are called internal observations since they are limited to traffic flow patterns and are agnostic to events outside the sensor network (i.e., external events). The second category is inclusive of events external to the sensor network monitoring the traffic flow. External events include active events (e.g., accidents, vehicle breakdowns) and scheduled events (e.g., sporting events, music events) [103; 61]. External events may be obtained by city authorities from sources such as 511.org for San Francisco Bay Area or by citizens in the city through social streams such as tweets.

2.2.2 Understanding Event Dynamics

We summarize related efforts in modeling time series observational data for understanding dynamical events.

2.2.2.1 General Approaches

Sustainability researchers are studying traffic conditions using sensors on road and GPS sensors on vehicles to predict congestion. Current research on traffic data analytics predominantly uses a single modality such as sensor data for understanding delays [74; 87; 12; 114; 40; 135]. Work on traffic diagnostics connects events to congestions utilizing historical data and applies it to extract the near real-time observations for explaining congestions in terms of city events [85; 37]. Inferring the root cause of traffic congestion is investigated by [32]. The origin and destination of a car is modeled as latent variables and the flow of cars observed from GPS data is modeled as the observed variable. The root cause does not include the city events that may influence traffic and even cause a change in the origin and the destination of cars. [61] use a Bayesian Network [77] structure extraction based approach to extract insights from a combination of traffic sensor data and incident reports. They derive insights that are not obvious to city authorities and present a traffic alert system to deliver these predictions to commuters.

2.2.2.2 Time Series Based Approaches

If we consider speed and travel time observations as time series observations, our proposition of explaining speed and travel time anomalies using textual observations can be viewed as a time series annotation task. In a related work [48], variations in the number of bicycles hired at various locations in a city, modeled as time series data, is explained using events in the city such as temporal landmarks, concerts, sports matches, parades, bad weather, and public holidays. Events are detected using location and date specific search queries to a search engine. An ensemble approach is used to build a model that connects major events to the number of bikes being hired. Annotating

physiological dynamics of premature babies for risk assessment has been carried out by [120], where, variations in physiological observations such as heart rate, blood pressure, and body temperature are modeled using a Switching Linear Dynamical System (SLDS).

Event interactions can be formulated utilizing statistical models to reveal the inherent interaction structure in a domain e.g., Bayesian Network structure extraction [35]. Relying on data alone for uncovering the statistical structure for understanding the interactions between events in a domain can be misleading [38; 39]. We propose techniques that can empower statistical models with declarative domain knowledge to understand the interactions between various events in a domain. Understanding real-world event dynamics utilizing both sensor and social data has not been explored adequately by the state-of-the-art approaches. We propose techniques to learn normalcy models from time series data and thereby determine anomalies. Further, we propose to explain the anomalies utilizing relevant events extracted from social data using spatio-temporal meta data.

2.3 Action Recommendation in PCS

We organize the work on action recommendation based on *one-shot* or *episodic*. One-shot action recommendation results in recommending a static sequence of steps (regardless of the changing information) analogous to a static map to navigate a city. Episodic action recommendation results in recommendations that are conducive to changing information analogous to a GPS to navigate a city. We will consider the domain of journey planning and AI planning techniques to summarize the related work on *one-shot* and *episodic* action recommendation respectively.

2.3.1 One-Shot Action Recommendation

There is a rich literature on journey planning, transit analysis and traffic events. Journey planners are available for prominent cities, e.g., San Francisco's 511 service [1]. A state-of-the-art algorithm for journey planning is described in [16]. However, such systems assume that real-time updates will

be available from sensors on vehicles. Google Transit provides a service for traveling in New Delhi but it is not adaptive and the details of how it gets data is not known since none of the city agencies provide their data in this format.

There are studies on public transit dealing with congestion and its impact on vehicle arrival [34]. But this line of work do not consider the influence of dynamic traffic events on public transport delays. Considering literature which has dealt with traffic events in a city, [63] studies the impact of an event on traffic patterns using simulation. The impact analysis is done over streets using cluster analysis but does not extend to public transportation. At a general level, [62] looks at traffic modeling and prediction using heterogeneous information sources but assumes a data-rich setting. Route recommendation based on traffic conditions is carried out by [62]. Our approach is novel in building an adaptive journey planner leveraging multi-modal textual updates (observations) for estimating delay probability and its impact on public transport schedule. We exploit prior knowledge of the domain to deal with data sparsity issues in cities with no instrumentation, i.e., no machine sensors to monitor the physical world.

2.3.2 Episodic Action Recommendation

Task driven computing [147; 99; 92] and AI Planning techniques and algorithms [117; 27; 23] can be utilized in providing assistance to users in reaching their goals such as carrying out DIY (Do-It-Yourself) tasks. Providing assistance within an Internet of Things and Services (IoTS) environment, however, faces new challenges, such as a constantly changing physical (e.g., available resources) and computing (e.g., available services) environment. AI Planning techniques model the problem of action recommendation using world states and transition functions. Classical AI Planning assumes enumeration of all world states and deterministic state transitions [51]. Hierarchical Task Network (HTN) planning is a widely used practical planning system [125; 140; 150] which imposes restrictions on non-primitive tasks and task ordering. Comparison of an HTN planner to the classical planning approaches along with expressiveness and complexity is presented in [45]. Planning under uncertainty

extends classical planning approaches to incorporate possibility of unreliable information, partially observable environment, and non-deterministic state transitions which captures the nature of the IoTS domain [28]. Action recommendation in the context of ubiquitous systems where a diverse set of resources and services are available in the ubiquitous environment is addressed by several research efforts [132; 116]. Some of the action recommendation problems are sequential where the next best action is to be recommended depending on the current state of the world. Such problems are characterized as sequential decision problems. Markov Decision Process (MDP) and Partially Observable Markov Decision Process (POMDP) are two such models for dealing with sequential decision making [21; 68]. Action recommendation systems may also utilize the *goodness of recommendation* based on user input to learn and adapt action recommendation. This area of research is also known as reinforcement learning [138].

One-shot action recommendation has been formalized and studied as part of Decision Theory [77; 13]. Episodic action recommendation has been studied extensively in the AI planning community [108]. Actions are recommended based on the initial state and the goal state by generating plans. PCS systems exhibits uncertainty and dynamism. Classical planning techniques that do not adapt the plan to reflect changing conditions may not achieve the "optimal" goal state. Planning under uncertainty accounts for dynamism and uncertainty of the domain to recommend actions [26]. Further, there may be declarative knowledge of actions in a domain which we can exploit for action recommendation. The state-of-the-art techniques in planning under uncertainty do not address the utilization of declarative knowledge of tasks and actions to initialize parameters of the recommendation model. We present techniques that utilize declarative knowledge of a domain in initializing parameters of a model for recommending next best action in the IoTS domain.

This dissertation has two major contributions. First, developing techniques to utilize both textual and sensor observations (multimodal) for understanding real-world events. Second, formulating the problems of event extraction, understanding, and action recommendation utilizing PGMs as a unifying formalism. Further, leverage declarative domain knowledge in learning structure and

parameters of PGMs.

3

Probabilistic Graphical Models

"Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering – uncertainty and complexity ..."

— Michael Jordan, UC Berkley, 1998

Probability theory originated in the context of dealing with uncertainty in playing games of chance. First systematic treatment of games of chance were provided by Gerolamo Cardano around mid 1500's in his book Liber De Ludo Aleae (Book on Games of Chance) published only by 1663 [112]. Probability theory provides a calculus to deal with uncertainty and has been utilized in various scientific fields. First axiomatic treatment of probability theory was carried out by Andrei Kolmogorov in his book published in 1933 [78]. Next important development in the field of probability theory was by Thomas Bayes, who proposed an approach to update probabilities upon receiving new information, referred to as the Bayes's Theorem. This theorem has survived the test of time [102] and has played a central role in solving many real-world problems, requiring reasoning from data under uncertainty spanning various domains such as medicine, rescue operations, military strategy, finance, and computer science. Graph theory was originated by Leonhard Euler's through his work published in 1736 on the feasibility of crossing The seven bridges of Königsberg [46] without

repeated traversal of any bridge by creating abstractions of the real-world. Since then, graph theory has been well studied, understood, and used for addressing complex real-world problems involving objects and their pairwise interactions.

Probabilistic Graphical Models (PGMs) utilize probability theory to deal with uncertainty and graph theory to deal with complexity, providing a unified framework to deal with two important problems in engineering. We will demonstrate the power of PGMs utilizing a medical use case we encountered in reducing asthma attacks in children. Alex wants to understand the reasons for her asthma exacerbation so that she can take preventive measures to reduce asthma attacks. Asthma is a multifaceted disease with high symptomatic variations across people. Understanding asthma exacerbations requires personal level signals (e.g., activity, medication compliance, wheeze level, cough), population level signals (e.g., asthma incidents reported at a location, report of asthma related symptoms), and environmental observations (e.g., air quality index, pollen level, presence of mold, humidity, temperature). Using this scenario of Alex, we will present some preliminaries of probability theory and then introduce Bayesian Networks which is a type of PGM.

3.1 Random Variables

Alex needs to represent these events and for simplicity, we assume that these events are binary. Pollen level (P), activity level in terms of number of steps (S), and medication taken (M) are some of the events of interest influencing the occurrence of asthma attacks (A). The upper case letters P, S, M, and A are called the random variables. Random variables are the variables whose value is not known until they are observed. For example, pollen in the environment may be *high* or *low*; we don't know the exact value till we observe the pollen content. Similarly, S is a random variable representing steps taken by Alex during a time duration which may be *high* or *low*. Alex can sometimes forget taking medication and this event is represented using the random variable M which can take values *yes* or *no*. Finally, Alex may or may not have an asthma attack represented by the random variable

A which can take values *yes* or *no*.

3.2 Probability Assignments

After defining random variables to capture the events of interest, Alex needs to observe various combinations of events represented as the value of random variables and assign probability of asthma attack. Formally, she needs to specify the joint probability distribution specified by $p(A, M, S, P)$ (more information on asthma management and its challenges on kHealth project page[1]). With four binary random variables, total number of probability assignments is $2^4 = 16$. Number of probability assignments to be done is exponential in the number of random variables and determining the required information is not tractable for real-world problems. The number of random variables in realistic models routinely exceed hundreds, leading to intractability if all the probability assignments has to be specified. There are two fundamental rules of probability that are utilized for inference in probabilistic graphical models. They are the *sum rule* and the *product rule* of probability as specified by (3.1a) and (3.1b) respectively. The X and Y are assumed to be independent random variables.

$$p(X) = \Sigma_Y p(X, Y) \tag{3.1a}$$

$$p(X, Y) = p(X|Y)p(Y) \tag{3.1b}$$

Applying equation (3.1b) repeatedly to the joint distribution $p(A, M, S, P)$ we obtain

$$p(A, M, S, P) = p(A|M, S, P)p(M|S, P)p(S|P)p(P) \tag{3.2}$$

3.3 Conditional Independence

Alex can utilize the domain knowledge of asthma management to address the challenge of enumerating all possible variable combinations. Alex does not have exercise induced asthma, consequently, the number of steps she takes has no influence on asthma attacks. Alex needs to take daily medication

[1] http://knoesis.org/projects/khealth

irrespective of the environmental conditions and her daily activity level. Thus, taking medication is independent of pollen in the environment and the number of steps she takes. Formally, these independences are represented as shown in Equation (3.3).

$$(A \perp S), (M \perp S), (M \perp P) \tag{3.3}$$

Applying these independences from Equation (3.3) to the joint distribution in Equation (3.2) we obtain the updated joint distribution given in Equation (3.4).

$$p(A, M, S, P) = p(A|M, P)p(M)p(S|P)p(P) \tag{3.4}$$

The conditional distribution $p(A|M, P)$ needs $2^2 = 4$ parameters, $p(S|P)$ needs 2 parameters, and $p(M)$ and $p(P)$ need one parameter each. Thus, the joint distribution with independences applied needs only $4+2+1+1 = 8$ parameters compared to 16 parameters without any embedded independence information. Domain knowledge plays a significant role in specifying independences among various random variables resulting in a significant reduction in the number of parameters to be specified. Probabilistic graphical models utilize *probability* to deal with uncertainty and *structure* to deal with complexity.

3.4 Bayesian Networks

The joint distribution specified by Equation (3.4) can be represented using a graphical model as shown in Figure 3.1 also referred to as a Bayesian Network (BN). Formally, a BN is a directed acyclic graph (DAG) with nodes representing random variables and edges representing dependencies (or independences) between them. The parameters of a BN is specified using a conditional probability table (CPT) for discrete random variables. For continuous random variables, the parameters are specified using a conditional probability distribution (CPD). Figure 3.1 contains parameters of the BN in the form of CPTs. The values in the CPTs of Figure 3.1 are for demonstration purposes only and this can be gleaned from real-wold data if available. Alex can pose questions related to her

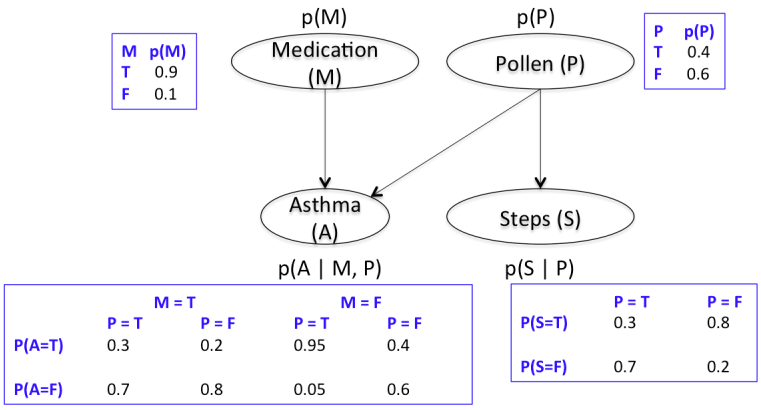


Figure 3.1: Dependencies (or independences) captured by a Bayesian Network for estimating the risk of asthma attacks

asthma attack risk over the BN. We will demonstrate two such queries: What is the probability that the pollen is high given that Alex had an asthma attack? What is the probability that the pollen is high given that Alex did not have an asthma attack? We show that the BN described here indeed satisfies our commonsense knowledge of asthma attacks.

The first question translates to the conditional probability query $p(P = T|A = T)$. We will utilize the joint distribution from Equation (3.4) and the parameters specified in Figure 3.1 to compute $p(P = T|A = T)$. Applying the product rule of probability described by the Equation (3.1b), we obtain Equation (3.5a). Applying the sum rule of probability from Equation (3.1a), we obtain Equation (3.5b). Expressing the joint distribution in terms of the conditional distribution, we obtain Equation (3.5c). In Equation (3.5c), $\Sigma_S p(S|P = T) = 1$ and $\Sigma_{S,P} p(S|P) = 1$ resulting in Equation (3.5d).

$$p(P = T|A = T) = \frac{p(P = T, A = T)}{p(A = T)} \tag{3.5a}$$

$$= \frac{\Sigma_{M,S} p(A = T, M, S, P = T)}{\Sigma_{M,S,P} p(A = T, M, S, P)} \tag{3.5b}$$

$$= \frac{\Sigma_M p(A = T|M, P = T)p(M)p(P = T)\Sigma_S p(S|P = T)}{\Sigma_{M,P} p(A = T|M, P)p(M)p(P)\Sigma_{S,P} p(S|P)} \tag{3.5c}$$

$$= \frac{\Sigma_M p(A = T|M, P = T)p(M)p(P = T)}{\Sigma_{M,P} p(A = T|M, P)p(M)p(P)} \tag{3.5d}$$

We look up values for these conditional distributions from the CPT in Figure 3.1 as shown in equation (3.6a).

$$= \frac{0.3 \times 0.9 \times 0.4 + 0.95 \times 0.1 \times 0.4}{0.3 \times 0.9 \times 0.4 + 0.2 \times 0.9 \times 0.6 + 0.95 \times 0.1 \times 0.4 + 0.4 \times 0.1 \times 0.6} \tag{3.6a}$$

$$p(P = T|A = T) = \frac{0.146}{0.278} = 0.5251 \tag{3.6b}$$

The probability that the pollen is high given that Alex had an asthma attack is given by the equation (3.6b), $p(P = T|A = T) = 0.5251$.

The second question is expressed as the probability of pollen being high given that Alex did not have an asthma attack, $p(P = T|A = F)$ as shown in equation (3.7a). We look up values for these conditional distributions from the CPT in Figure 3.1 as shown in equation (3.7b).

$$= \frac{\Sigma_M p(A = F|M, P = T)p(M)p(P = T)}{\Sigma_{M,P} p(A = F|M, P)p(M)p(P)} \tag{3.7a}$$

$$= \frac{0.7 \times 0.9 \times 0.4 + 0.05 \times 0.1 \times 0.4}{0.7 \times 0.9 \times 0.4 + 0.8 \times 0.9 \times 0.6 + 0.05 \times 0.1 \times 0.4 + 0.6 \times 0.1 \times 0.6} \tag{3.7b}$$

$$p(P = T|A = F) = \frac{0.254}{0.722} = 0.3518 \tag{3.7c}$$

The probability that the pollen is high given that Alex did not have an asthma attack is given by the equation (3.7c), $p(P = T|A = F) = 0.3518$. Asthma attacks are likely when the pollen in the environment is high, and conversely, if we know Alex had an asthma attack, we expect the pollen to be high. If we know Alex did not have an asthma attack, it is less likely that the pollen is high. This intuition is satisfied by our computation resulting in $p(P = T|A = T)$ from equation (3.6b) is greater than $p(P = T|A = F)$ from equation (3.7c).

3.4.1 Factors

A factor is a function that maps random variable assignments to a real number. Factors are used as a unifying representation for both directed (e.g., Bayesian Networks) and undirected (e.g., Markov Networks) graphical models. Representation of a probabilistic graphical model along with the factors

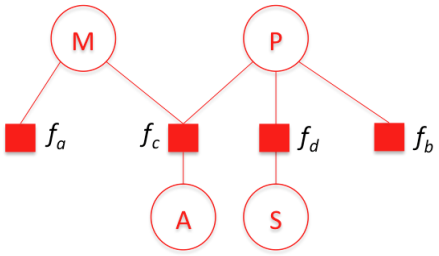


Figure 3.2: Factor graph representation of the Bayesian Network in Figure 3.1 where, M, P, A, and S represent the random variables for medication, pollen, asthma, and steps respectively

is called a *factor graph*. Factor graph include explicit representation of the function as a node that, maps random variable assignments to the corresponding real values. These real values can be probabilities (normalized) or potentials (un-normalized). Scope of a factor is a set of all the random variables that appear in a factor. The Bayesian Network in Figure 3.1 can be represented using explicit factors $f_a = \phi(M)$, $f_b = \phi(P)$, $f_c = \phi(A, M, P)$, and $f_d = \phi(S, P)$ as shown in Figure 3.2. Construction and inference in probabilistic graphical models can be posed as operations on factors.

$$p(\mathbf{X}) = \prod_s f_s(\mathbf{X_s}) \tag{3.8}$$

In general, a joint distribution over a set of random variables $\mathbf{X}$ can be denoted by a product of factors over a subset of variables denoted by $\mathbf{X_s}$. The joint distribution in equation (3.4) can be obtained by factor product using equation (3.8) as $p(A, M, S, P) = p(M)p(P)p(A|M, P)p(S|P) = f_a \times f_b \times f_c \times f_d = \phi(M) \times \phi(P) \times \phi(A, M, P) \times \phi(S, P)$. *Factor product*, *factor marginalization*, and *factor reduction* are the three basic operations on factors for constructing any arbitrary network and perform inference over the constructed network. *Factor product* can be used to multiply various factors for creating a joint distribution over random variables. *Factor marginalization* performs summing over given variables effectively eliminating those variables. *Factor reduction* modifies the assignment of values or potentials to make it consistent with the observed evidence.

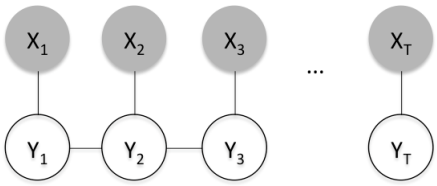


Figure 3.3: A Conditional Random Field (CRF) model with observed variables $\mathbf{X}$ and the target variables $\mathbf{Y}$

3.5 Conditional Random Fields

Conditional Random Fields (CRFs) have been extensively used in text processing specifically for part-of-speech tagging and named entity recognition [82]. CRFs are a type of undirected probabilistic graphical models containing two types of variables; observed variables denoted by $\mathbf{X}$ and the target variables denoted by $\mathbf{Y}$. The nodes in the CRF graph corresponds to $\mathbf{X} \cup \mathbf{Y}$. A CRF models the conditional distribution $p(\mathbf{Y}|\mathbf{X})$ factorizing over the graph shown in Figure 3.3. A CRF model does not allow the probability distribution over the observed variables $\mathbf{X}$ alone. This constraint of a CRF model is beneficial in situations where we do not really know the dependencies between the observed variables $\mathbf{X}$.

Consider a CRF model as shown in Figure 3.3 with $\mathbf{Y} = \{Y_1, ..., Y_T\}$ and $\mathbf{X} = \{X_1, ..., X_T\}$ containing potentials between (Y_i, Y_{i+1}) and (Y_i, X_i). Representing the potentials using a factor graph representation as introduced in sub-section 3.4.1, we obtain the factor graph shown in Figure 3.4. The potentials between adjacent target variables (Y_i, Y_{i+1}), f_b and f_d in Figure 3.4 are defined by potential of the form $\phi(Y_i, Y_{i+1})$, i.e., $f_b = \phi(Y_1, Y_2)$ and $f_d = \phi(Y_2, Y_3)$. The potentials between the target and observed variable (Y_i, X_i), f_a, f_c, f_e, and f_t are defined by potentials of the form $\phi(Y_i, X_i)$, i.e., $f_a = \phi(Y_1, X_1)$, $f_c = \phi(Y_2, X_2)$, $f_e = \phi(Y_3, X_3)$, and $f_t = \phi(Y_T, X_T)$. The conditional distribution $p(\mathbf{Y}|\mathbf{X})$ over all the observed and target variables $\mathbf{X} \cup \mathbf{Y}$ is given by equation (3.9a) where, $\tilde{p}(\mathbf{Y}, \mathbf{X})$ consists of unnormalized potentials and $Z(\mathbf{X})$ is a normalizing constant. The unnormalized distribution $\tilde{p}(\mathbf{Y}, \mathbf{X})$ can be expressed in the form of factor product shown in equation (3.9b)

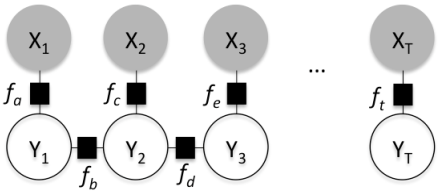


Figure 3.4: A factor graph representation of the Conditional Random Field (CRF) model with factors representing potentials between the observed variables $\mathbf{X}$ and the target variables $\mathbf{Y}$

and (3.9c). Representing the factor product in a concise manner results in equation (3.9d) and the normalization constant is computed using equation (3.9e).

$$p(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})}\tilde{p}(\mathbf{Y}, \mathbf{X}) \tag{3.9a}$$

$$\tilde{p}(\mathbf{Y}, \mathbf{X}) = (f_a.f_c.f_e.....f_t).(f_b.f_d....) \tag{3.9b}$$

$$\tilde{p}(\mathbf{Y}, \mathbf{X}) = (\phi(Y_1, X_1).\phi(Y_2, X_2).\phi(Y_3, X_3).....\phi(Y_T, X_T)).(\phi(Y_1, Y_2).\phi(Y_2, Y_3)....) \tag{3.9c}$$

$$\tilde{p}(\mathbf{Y}, \mathbf{X}) = \prod_{i=1}^{T-1} \phi(Y_i, Y_{i+1}) \prod_{i=1}^{T} \phi(Y_i, X_i) \tag{3.9d}$$

$$Z(\mathbf{X}) = \sum_{Y} \tilde{p}(\mathbf{Y}, \mathbf{X}) \tag{3.9e}$$

Given training examples with observed variables and target variables, a CRF model can be utilized to learn the distribution $\tilde{p}(\mathbf{Y}, \mathbf{X})$ using Maximum Likelihood or Stochastic Gradient Methods [137]. We will find the target variable assignment $\mathbf{Y}$ that results in the highest value for the conditional probability $p(\mathbf{Y}|\mathbf{X})$ given the observed variables $\mathbf{X}$.

3.6 Linear Dynamical Systems

Linear Dynamical Systems (LDS) [69], also known as Kalman Filters [70] in the context of signal processing, are widely utilized in applications spanning astronomy, physics, control systems, econometric system modeling, surveillance, and object tracking. Time series data with hidden and observed variables naturally occur in these domains. A LDS model [15] incorporates both hidden

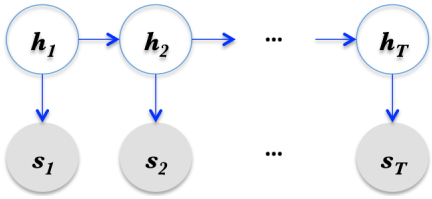


Figure 3.5: A Linear Dynamical System for T time points with hidden nodes $h_{1:T}$ and observed nodes $s_{1:T}$.

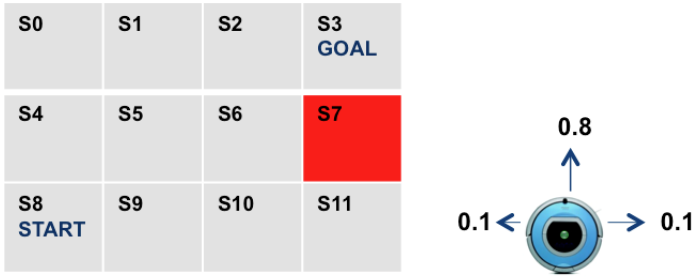
and observed variables as shown in Figure 3.5 with T hidden nodes $h_{1:T}$ and T observed nodes $s_{1:T}$ respectively for modeling observations at T time points. A hidden variable captures the state of a system that is not directly observable, e.g., in the context of diseases and symptoms, a disease is hidden while a symptom is observable. In the domain of traffic, the volume of vehicles passing through a link may not be available (511.org does not provide volume data). Further, there may be many other unobserved factors influencing traffic dynamics such as road conditions, visibility, and random effects. These unobserved variables at time t may be represented using a hidden node h_t in the LDS model. The average speed of vehicles and average travel time through a link are the observed variables represented using s_t in the LDS model. LDS is formally defined using Equations(3.10a) and (3.10b) where $\mathbf{A_t}$ is called the *transition matrix* and $\mathbf{B_t}$ is called the *emission matrix*. $\eta_{\mathbf{t}}^{\mathbf{h}}$ and $\eta_{\mathbf{t}}^{\mathbf{s}}$ represent the transition and emission noise, respectively.

$$\mathbf{h_t} = \mathbf{A_t h_{t-1}} + \eta_{\mathbf{t}}^{\mathbf{h}}, \quad \eta_{\mathbf{t}}^{\mathbf{h}} \sim \mathcal{N}(\eta_{\mathbf{t}}^{\mathbf{h}} | \bar{\mathbf{h}}_{\mathbf{t}}, \mathbf{\Sigma}_{\mathbf{t}}^{\mathbf{h}}) \tag{3.10a}$$

$$\mathbf{s_t} = \mathbf{B_t h_t} + \eta_{\mathbf{t}}^{\mathbf{s}}, \quad \eta_{\mathbf{t}}^{\mathbf{s}} \sim \mathcal{N}(\eta_{\mathbf{t}}^{\mathbf{s}} | \bar{\mathbf{s}}_{\mathbf{t}}, \mathbf{\Sigma}_{\mathbf{t}}^{\mathbf{s}}) \tag{3.10b}$$

The hidden state at any time, $\mathbf{h_t}$, depends only on the previous hidden state $\mathbf{h_{t-1}}$ (Markovian assumption) and the transition from $\mathbf{h_{t-1}}$ to $\mathbf{h_t}$ is governed by the *transition matrix*. The observation at any time, $\mathbf{s_t}$, depends only on the current hidden state $\mathbf{h_t}$ and is governed by the *emission matrix*. Their joint probability distribution over all the hidden states and observations is given by

$$p(h_{1:T}, s_{1:T}) = p(h_1)p(s_1|h_1) \prod_{t=2}^{T} p(h_t|h_{t-1})p(s_t|h_t) \tag{3.11}$$



(a) (b)

Figure 3.6: (a) Stochastic environment which contains $4 \times 3 = 12$ states. (b) A robot a.k.a an agent in the stochastic environment with 80% probability of movement along the instructed direction and 20% probability of moving perpendicular to the intended direction.

where, the terms $p(h_t|h_{t-1})$ and $p(s_t|h_t)$ are given by

$$p(h_t|h_{t-1}) = \mathcal{N}(h_t|A_t h_{t-1} + \bar{h}_t, \Sigma_t^h) \tag{3.12a}$$

$$p(s_t|h_t) = \mathcal{N}(s_t|B_t h_t + \bar{s}_t, \Sigma_t^s) \tag{3.12b}$$

This model offers to capture variations in the form of transition and emission matrices with a Gaussian noise. For the domain of traffic, we assume that the transition and emission matrices do not vary over time. Such a model is called a stationary model. Thus, $\mathbf{A_t} \equiv \mathbf{A}$, $\mathbf{B_t} \equiv \mathbf{B}$, $\mathbf{\Sigma_t^h} \equiv \mathbf{\Sigma_h}$, $\mathbf{\Sigma_t^s} \equiv \mathbf{\Sigma_s}$, $\mathbf{\bar{h}_t} = 0$, and $\mathbf{\bar{s}_t} = 0$. The hidden state $\mathbf{h_t}$ is normally distributed with mean $\mathbf{A_t h_{t-1}}$ and covariance Σ_h. The observation $\mathbf{s_t}$ is normally distributed and has mean $\mathbf{B_t h_t}$ and covariance Σ_s.

For the problem of predicting the future observations, we need to first learn the transition ($\mathbf{A_t}$) and emission matrices ($\mathbf{B_t}$) from the training time series data. Since LDS parameter learning involves hidden variables, EM algorithm [54] can be utilized to estimate $\mathbf{A_t}$ and $\mathbf{B_t}$. Later, we could utilize $\mathbf{A_t}$ and $\mathbf{B_t}$ to predict observation at $t+1$ utilizing equations(3.10a) and (3.10b).

3.7 Markov Decision Process

Probabilistic graphical models discussed till now deal with various challenges such as uncertainty, complexity, and dynamism in PCS systems. We will now introduce a probabilistic graphical model that formalizes the problem of decision making in stochastic environment using an example borrowed from [124]. Stochasticity is inherent in many real-world problems and we demonstrate such an environment in Figure 3.6(a). There are twelve possible locations which we also call states. A robot or an agent shown in Figure 3.6(b) needs to navigate from start state to the goal state shown in Figure 3.6(a). Robot can receive and execute commands *[Up, Down, Right, Left]* to navigate in the environment. When the robot receives a command, the probability that it succeeds in ending up in the desired state is 0.8 for all the commands *[Up, Down, Right, Left]*. For example, if the robot is in state S5, an *Up* command issued will leave the robot in state S1 with 0.8 probability. The robot may end up in state S4 with a probability of 0.1 or it may end up at state S6 with a probability of 0.1. In summary, the robot moves in the intended direction with a probability of 0.8 but may move in perpendicular direction with a 0.2 probability value. In an ideal deterministic world, the sequence of instruction to navigate the robot from start state S8 to goal state S3 is *[Up, Up, Right, Right, Right]*. However, in the stochastic environment of Figure 3.6(a), the probability of the robot reaching the goal state following the provided commands is estimated as $0.8^5 = 0.32768$.

In an environment where there is an initial state and a goal state, we need a mechanism to specify the notion of utility of each state. Utility of a state is defined for each state. Attaining a goal state may involve taking multiple intermediate states. The utility of each intermediate state is accumulated when progressing toward the goal state. Various states and associated utility is shown in Figure 3.7. The goal state has a utility of +1 and rest of the states have a utility of -0.04 except for one undesirable state (shown in red in Figure 3.7) that has a utility of -10. With this utility configuration, if the robot takes ten steps to reach the goal state, then, the aggregate utility is 0.6 ($= 10 \times -0.04 + 1$). The robot has no incentive staying in any state except for the goal state and

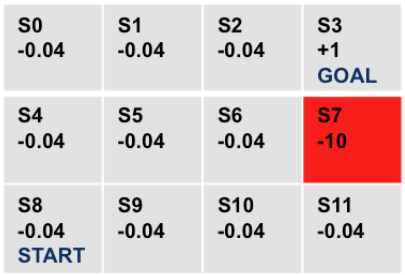


Figure 3.7: Stochastic environment with rewards for each state annotated within the box representing the states. The goal state has a reward of +1, undesired state has a reward of -10, and all other states have a reward of -0.04.

hence the best aggregate utility is achieved with those paths that are shortest to the goal state. We observe that this is a sequential decision problem which is fully observable with a Markovian transition and additive rewards. Such a formalism is called a Markov Decision Process (MDP).

We observe that the deterministic solution to navigate the robot to the goal state from the start state does not guarantee the desired final state. To address this non-determinism, we need an action recommendation at each state and such a definition of action in each state is called a *policy* denoted by π. If s is the current state, then $\pi(s)$ is the action recommended by policy π in state s. A policy that results in the highest expected utility is called an optimal policy π^*. Since the rewards are additive, the expected utility is given by equation (3.13).

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \tag{3.13}$$

Where, U^{π} is the expected utility of a policy π starting from state s. γ^t is the discounting factor, S_t is the random variable representing the state that the agent would attain at time t, and $R(S_t)$ is the utility value at state S_t. The expected utility U^{π} provides the desirability of utilizing a certain policy π. Actions taken in the distant past should not influence the current rewards heavily and this is seldom the case in problems with additive rewards. The discounting factor γ^t enables us to minimize the undesirable impact of actions in the distant past on the current rewards.

The notion of best policy is formally defined by the function given by equation (3.14).

$$\pi^* = \underset{b}{\operatorname{argmax}}\, U^{\pi}(s) \tag{3.14}$$

Value iteration and Policy iteration [49] are two important techniques to derive the optimal policy which can be later utilized by an agent for sequential decision making in a stochastic environment. If there are b policies, the optimal policy is defines as the policy with the highest expected utility U^{π} defined by the argmax operation.

All the probabilistic models introduced in this chapter – Bayesian Networks (BNs), Conditional Random Fields (CRFs), Linear Dynamical Systems (LDS), and Markov Decision Process (MDP) are a type of probabilistic graphical model. Throughout the dissertation, we will be utilizing the probabilistic graphical models introduced in this chapter to formalize problems related to event extraction, event understanding, and action recommendation in PCS systems.

4

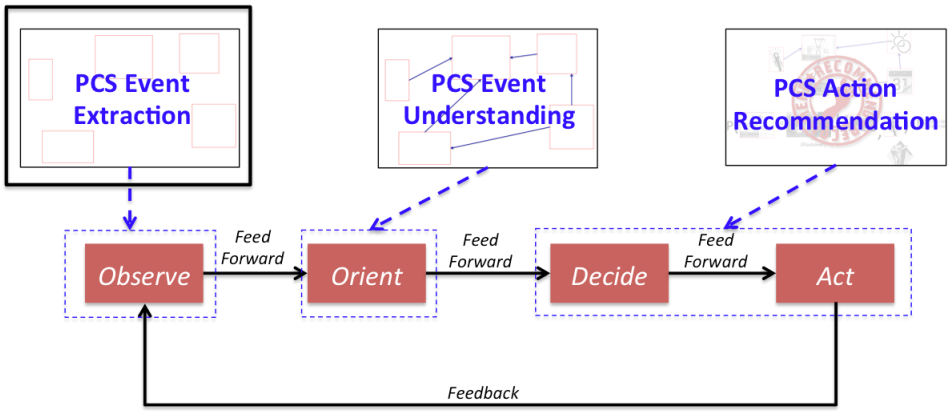
Event Extraction

"nothing drives basic science better than a good applied problem"

— Newell & Card, p. 238, 1985

As started by the 1975 Turing Award winner, Allen Newell, and Stuart K. Card, we believe that an applied problem of extracting events related to city traffic events will pave the way for solving some challenging problems in the theory of learning probabilistic graphical models. This choice of traffic domain is not only based on the availability of open data and ground truth for evaluation purposes but also for its growing importance as we move forward. Cities are a good example of physical, cyber, social (PCS) Systems. Cities have been a thriving place for citizens over centuries due to a range of socio-economic opportunities. By 2001, over 285 million people lived in cities of India, which was more than the population of the entire United States then [119]. This trend of citizens moving to cities is creating tremendous pressure on the city infrastructure. Understanding the status and interactions between city systems is crucial to enable smooth functioning of a city. City authorities face numerous challenges in deploying, maintaining, and optimizing operations and interactions between various city departments and services (collectively called city infrastructure). They are also pressed for ways to minimize wastage of resources, improve efficiency, and be economically self-sustaining. Understanding city events is of great contemporary interest [107; 89; 72] emphasizing the

Figure 4.1: Relationship between components of OODA-loop and components of PCS-Analytic framework



crucial need for extracting and analyzing city events. Citizen sensing [129; 30] component that may provide complementary or corroborative information is often ignored in the current state-of-the-art analytics for Smart Cities [52; 93]. We show that social data streams harnessed in the context of Smart Cities provide a comprehensive view of events in a city complementing other modalities such as observations from city authorities.

Twitter, a microblogging platform, has developed into a near real-time source of information spanning heterogeneous topics of varying importance. With over 500 million users world-wide, twitter generates 500 million tweets a day[1]. Increasingly, tweets have been a source of interesting and vital information such as status of public transport, traffic and environmental conditions, public safety, and general events in a city.

As explained earlier in the introduction, PCS event extraction aligns with the *observe* step of OODA-loop as shown in Figure 4.1. We address the following research questions in this chapter: How do we extract city traffic infrastructure related events from tweets? How can we exploit event and location knowledge-bases for event extraction? How well can we extract city traffic events? How can we extract traffic related events from SMS messages? In addressing these questions, we outline the challenges in extracting events related to city infrastructure from informal text, and

[1]http://bit.ly/1TLUo8H

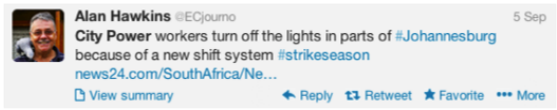
demonstrate the effectiveness of our approach through a comprehensive evaluation in the context of traffic related events. Specifically, we compare with ground truth provided by 511.org traffic incident reports showing the promise of our approach.

The following are the contributions of our work: i) We hypothesize and validate the role of citizen sensing in extracting city traffic events. ii) We develop an automatic training data creation process to train the annotation model by utilizing existing event and location knowledge-bases. iii) We design and implement a city event extraction algorithm to extract events from annotated (event and location terms) textual data. iv) We evaluate our approach concretely by comparing events extracted from social streams and events reported on 511.org using three orthogonal metrics to emphasize their complementary, corroborative, and timely nature.

4.1 Textual Data for Real-time Updates in PCS Systems

We illustrate the role of citizen sensing for understanding city infrastructure related events, and present related work on event extraction.

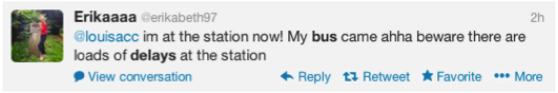
Typically, a city has many departments such as public safety, urban planning, energy and water, environmental, transportation, social programs, and education [19; 20]. Some of the services offered by these departments are dynamic, e.g., transportation services and their offerings may vary in response to sporting and music events, accidents, and weather conditions. Timely understanding of the situation is important for city authorities to manage city resources. Figure 4.2 depicts real-world city events reported directly by citizens in near real-time on twitter. They relate to power outages, poor water quality, a procession, and a delay experienced on public transport system as depicted in Figure 4.2(a), 4.2(b), 4.2(c), and 4.2(d) respectively. This information complements sensor data or textual data from conventional sources such as city departments. For example, sensors deployed on a road may report reduced speed of vehicles which can be explained by a procession obstructing traffic.



(a) Power shutdown

(b) Poor water quality

(c) Traffic jam due to procession



(d) Delay in public transit system

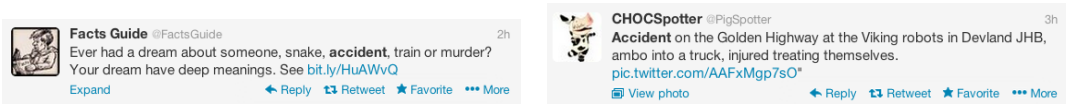
Figure 4.2: Tweets reporting various concerns about a city, which spans power supply, water quality, traffic jams, and public transport delays

In many cities around the world, there is an immense pressure on city infrastructure. Fine-grained sensor data may not be available from such cities due to the lack of extensive instrumentation. Citizen sensing can play a crucial role in filling the void in such environments [9].

4.2 Traffic Event Extraction from Tweets

Current event extraction techniques use event specific patterns based on event types [139; 56]. The text is expected to have some structure (e.g., news documents). Such a technique does not scale for city events from twitter text due to the informal nature of tweets. Further, the aggregation is done at a cluster level which is too coarse grained for city related events. E.g., important city events may be reported by few citizens given the wide scope of topics on twitter [80]. In such situations, clustering based techniques [109] may fail to separate minority tweets related to the city infrastructure.

Twitter messages may be noisy and convoluted with little context in which the message has been generated. Such a characteristic of tweets challenges a dictionary based approach for spotting location and event terms. For example, Figure 4.3 shows one of many instances where a single term 'accident' is used in different contexts. The tweet in Figure 4.3(a) refers to the dream a person had while the tweet in Figure 4.3(b) actually refers to an accident. The tweet in Figure 4.3(a) does not contain any location information while the tweet in Figure 4.3(b) has "Golden Highway"



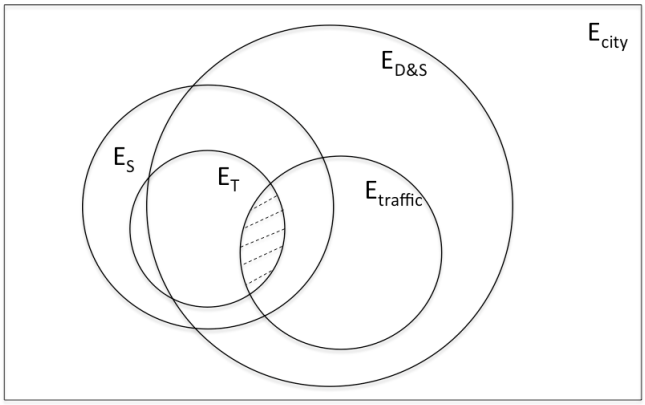
(a) Accident in the context of a dream (b) Accident in the context of a road accident

Figure 4.3: Addressing ambiguity: challenge for event extraction - tweets reporting very different events using the same term 'accident'

as the location. It is clear that relying solely on a dictionary based context free spotting of event terms cannot capture these nuances due to context-sensitive dependencies between words. It is possible to use a purely dictionary based approach for spotting event terms but would require human inspection for accurate tagging. Such manual inspection of the results of event spotting is infeasible because of the volume and velocity of the tweets, and the need for quick action. In order to automate this process, we formalized this problem of spotting event terms and location names as a sequence labeling problem. We then evaluated the performance of dictionary based spotting of event and location terms for a relative comparison with sequence labeling models. We provide some insights on the benefits of using dictionary based approaches for creating training data instead of directly using the dictionary for entity spotting. In practice, the training data may require some cleaning depending on the required accuracy of spotting and availability of resources. Our city event extraction framework provides control over the manual effort required to clean the training data. We create a training set for building a Conditional Random Field (CRF) [81] model automatically, by using dictionary-based spotting, to reduce manual tagging effort. We organize the details of our approach into event annotation and extraction.

Our approach is motivated by the open domain event extraction from twitter [122]. We introduce basic notations used in this chapter and elaborate on the solution components.

Figure 4.4: Depiction of city events (E_{city}, $E_{D\&S}$, and $E_{traffic}$) and its relationship to city events from social streams (E_S) and twitter (E_T)



4.2.1 Preliminaries

Figure 4.4 summarizes the relationship between events in a city and citizen observations using Venn diagram. Let E_{city} be the set of city events. Let $E_{D\&S}$ be the subset of events related to the city infrastructure (departments and services offered in a city) $E_{D\&S} \subset E_{city}$. $E_{D\&S}$ is not directly observed but we have access to the social streams represented by S containing events E_S. E_S may contain a subset of events related to city department and services represented as $E_S \cap E_{D\&S}$. City events flow through two major information channels: formal reporting and informal reporting. In formal reporting, dedicated resources such as machine sensors or city department officials observe and report various city events. Citizens may report their observations of a city through location based services (e.g., foursquare[2]), event based services (e.g., eventful), and user generated content (e.g., blogs, posts, and tweets). We focus our attention on events from twitter stream, which have been widely accepted as a near real-time source of citizen chatter [106] about traffic events. The shaded region in Figure 4.4 represents traffic events from twitter. We represent the traffic events extracted from twitter by the set E_T. Specifically, we use E_T to present and evaluate our algorithms and techniques. We use $E_{traffic} \subset E_{D\&S}$ obtained from 511.org as ground truth due to its open

[2]https://foursquare.com/

Figure 4.5: A sample tag assignment to tokens (words) in a tweet where B-EVENT indicates beginning of an event entity, B-Location and I-Location indicates beginning and intermediate words or last word of a location entity, and O is used to label non-entity words

Accident B-EVENT on O the O Golden B-LOCATION Highway I-LOCATION at O the O Viking O robots O in O Devland O JHB O , O ambo O truck O , O injured B-EVENT treating O themselves O

availability.

4.2.2 Basic Notations

We define the event schema as a 5-tuple $\langle \hat{e}_{type}, \hat{e}_{loc}, \hat{e}_{st}, \hat{e}_{et}, \hat{e}_{impact} \rangle$ where $\hat{e}_{type}$ ranges over all the events in E_T extracted from twitter T. $\hat{e}_{type}$ refers to the event type such as accident, breakdown, and music event, $\hat{e}_{loc}$ refers to the location of the event (lat-long), $\hat{e}_{st}$ and $\hat{e}_{et}$ refers to the start time, and the end time of the event, and $\hat{e}_{impact}$ refers to a number quantifying the severity of the event. We use hat ($\hat{}$) to emphasize that these are estimated values as actual values are unavailable. When available, the ground truth of events from city authorities may be used as actual values.

4.2.3 Problem Formulation

We formulate the problem of detecting events from informal text as a sequence labeling and aggregation problem.

4.2.3.1 Annotation

A tweet is composed of a sequence of tokens, $tokens(tweet_n)$ where *tokens* is a function that emits tokens given input tweet, $tweet_n$. A tag is a label given to each token in the token sequence $tokens(tweet_n)$. To annotate multi-phrase entities, we use a variant of the widely accepted BIO notation [121] in computational linguistics[3]. For a single word entity, we use the B- suffix/label. For a multi-word phrase, we use the B- suffix for the first word and I- suffix for all the subsequent

[3]http://en.wikipedia.org/wiki/Inside_Outside_Beginning

words including the last word. If a word does not refer to an entity it is suffixed with O. Using BIO notation to annotate a location entity phrase *Golden Highway*, we get *Golden B-Location Highway I-Location*. Entities that are not related to events and locations are tagged as Other (O). In general, the tag set contains location tags (B-Location, I-Location) and event tags (B-Event, I-Event), Tag_{set} = { B-Location, I-Location, B-Event, I-Event, O }. We want to assign the most relevant tag to each token in $tokens(tweet_n)$ taking into account dependencies between tokens, e.g., phrase named entities and long distance dependencies. For example, occurrence of 'accident' along with a location name vs. 'dream' without location names as shown in Figure 4.3. Figure 4.5 shows a sample tag assignment for tokens in a tweet.

4.2.3.2 Extraction

Once we have the most likely tag assigned to each token in a tweet, we proceed to perform city traffic related event extraction. One event quintuple is synthesized from a collection of tweets originating from a close neighborhood (or a small region) and in a short span of time. Highly informal, redundant, and noisy nature of tweets requires us to rank and aggregate events based on location, time, and theme dimensions as detailed in the next section. Aggregation algorithm summarizes redundant report of events and creates a unique representation $\langle \hat{e}_{type}, \hat{e}_{loc}, \hat{e}_{st}, \hat{e}_{et}, \hat{e}_{impact} \rangle$ for each event by grouping quintuples $\langle n, e, t, d, l \rangle$ based on location, time, and theme dimension. As a pre-processing step, before emitting the event tuple $\langle \hat{e}_{type}, \hat{e}_{loc}, \hat{e}_{st}, \hat{e}_{et}, \hat{e}_{impact} \rangle$, we emit a quintuple $\langle n, e, t, d, l \rangle$ for each tweet collected from a city where n represents location terms, e consists of the event terms, t represents the time of day, d represents the day of week, and l represents the geohash location. We run aggregation algorithms on this representation to uncover events in an unsupervised manner. We cluster tuples $\langle n, e, t, d, l \rangle$ for deriving the event tuple, $\text{e}_i \in E_T$, based on event terms and then filter based on geohash location. Event terms that we spot in tweets directly map to event types (syntactically) in a comprehensive hierarchy of events provided by 511.org[4]. This mapping is due to the

[4]Metropolitan Transportation Commission, http://511.org

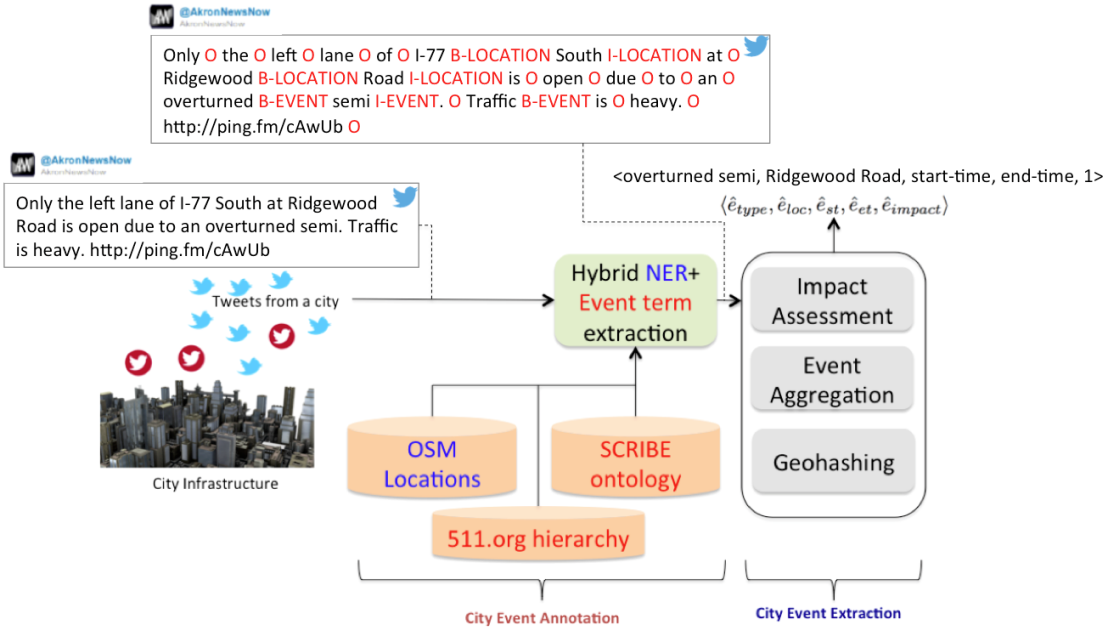


Figure 4.6: Architecture for extracting city infrastructure related events from social stream such as tweets

use of training data (to train the CRF model) containing event types from the same hierarchy. This hierarchy has *active-events* and *scheduled-events* as two major categories. Unlike conventional event extraction from text, city events require aggregation algorithms to be strongly tied with location (space) and time.

4.2.4 Solution Components

The tweet processing pipeline is shown in Figure 4.6 and the details of each solution component is presented here. Location and time are crucial components for city related event extraction. We are exploiting spatio-temporal context/coherence for city event extraction from informal text.

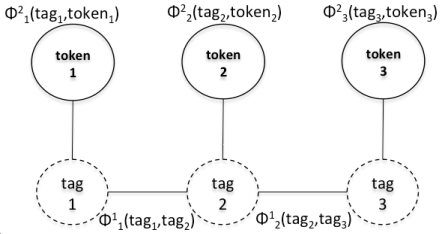
4.2.4.1 City Event Annotation

A CRF model is an undirected graphical model [77; 44; 33] containing nodes that correspond to the set: $tokens(tweets_n) \cup \text{Tag}_{set}$. The model defines factors to capture dependencies between (a) neigh-

boring tags (tag_i, tag_{i+1}) and (b) tags and tokens sequence $(tag_1, token_1)$,...,$(tag_i, token_i)$,...,$(tag_m, token_m)$ where $tag_i \in Tag_{set}$ and $token_i \in tokens(tweets_n)$. A factor is a function that maps all possible values of input variable combinations to real numbers where $V \subset tokens(tweets_n) \cup \text{Tag}_{set}$. This number is also called the potential for the input variable combinations, e.g., $\phi(tag_i, tag_{i+1})$ captures the number of times tag_i appears before tag_{i+1} in a corpus. Concretely, if tag_i is B-Location and tag_{i+1} is I-Location, ϕ(B-Location, I-Location) maps to the number of times this sequence appears in the corpus. This may not be a normalized value. If $tokens(tweets_n) = \{token_1, token_2, ..., token_m\}$, we define factors $\phi(tag_i, token_i)$ for each token where the $token_i$ is always observed. This factor captures the number of times the token $token_i$ was labeled with the tag tag_i. Concretely, if $token_i$ is *Golden* and tag_i is B-Location, then ϕ(B-Location, Golden) captures the number of times the token *Golden* was labeled with the tag B-Location in the corpus. A simplified example of the model is shown in Table 4.1. If there are m tokens in a sequence, we need (m - 1) factors to define potentials between neighboring tags and m factors to define potentials between tags and tokens.

Finding the most likely tag assignment can be formalized as maximizing the probability $P(tags|$ $tokens(tweet_n))$ shown in Table 4.1(a). $\widetilde{P}(tags, tokens)$ is the unnormalized score for a configuration of tokens and its tag assignment represented by *tags*. The term *argmax* selects the tag assignment for all the tokens based on the highest probability score. Even though the model captures potentials between adjacent tags, tag assignment is done based on the global maximum, i.e., tags that result in highest overall score are assigned to all the tokens. Such a global assignment of tags naturally captures long distance dependencies in text. The location and event spotting model use the linear chain CRF model presented in Table 4.1(b) and implemented by LingPipe[5]. The LingPipe implementation of CRF uses a slightly different model compared to the simplified one in Table 4.1(a). Both the CRF models in Tables 4.1(a) (b) are linear chain CRFs. Linear chain CRF restricts the factors to be defined only between adjacent tags. Arbitrary tag dependencies are not allowed in the CRF model. The CRF model also disallows joint distribution among the tokens and models only the conditional distribution between the tag and the tokens i.e., $P(tags|tokens(tweet_n))$. But the

Table 4.1: Formalization of sequence labeling task using a Conditional Random Field (CRF) on the left and LingPipe CRF implementation on the right



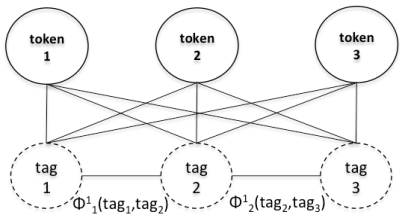
(a)

$P(tags|tokens) = \frac{1}{Z(tokens)} \widetilde{P}(tags, tokens)$

$\widetilde{P}(tags, tokens) = \prod_{i=1}^{m-1} \phi(tag_i, tag_{i+1}) \prod_{i=1}^{m} \phi(tag_i, token_i)$

$Z(tokens) = \sum_{tags} \widetilde{P}(tags, tokens)$

$argmax_{tags \in Tag_{set}} P(tags|tokens(tweet_n))$



(b)

$tokens = x[1], ..., x[N]$

$tags = 1, ..., m$

$F = f(pos = i, prevTag = j) \forall i = 1, j = null \ and \ i = 2, ..., N; j = 1, ..., m$

$\beta[m] < -LogisticRegression(F)$

$\widetilde{P}(tags, tokens) = \phi(tag_1, ..., tag_N | x[1], ..., x[N])$

$p(tag_1, ..., tag_N | x[1], ..., x[N]) = \frac{\phi(tag_1, ..., tag_N | x[1], ..., x[N])}{Z(x[1], ..., x[N])}$

$\phi(tag_1, ..., tag_N | x[1], ..., x[N]) = f(1, null) \cdot \beta[tag_1] + f(2, tag_1) \cdot \beta[tag_2] + ... + f(N, tag_{N-1}) \cdot \beta[tag_N]$

$Z(x[1], ..., x[N]) = \sum_{tag_1=1:m} ... \sum_{tag_N=1:m} \phi(tag_1, ..., tag_N | x[1], ..., x[N])$

tags may depend on any arbitrary feature extracted from the sequence of *tokens*. Each tag type and its positions in a corpus are extracted using a feature extractor function $f \in$ F which takes current token position and tag assigned to the previous token as input. The first token in the sequence will have *null* as the previous tag. For rest of the tokens in the input sequence, the feature function is invoked with all possible tags (1,..,m). $\beta[m]$ are the coefficient vectors learned for each output tag in the tag set T_{tags} where m is the number of tags from the corpus in the training phase. $\beta[m]$ is learned using Logistic Regression in the LingPipe implementation. The corresponding unnormalized score for tag assignment given tokens represented as $\phi(tag_1, ..., tag_N | x[1], ..., x[N])$ is computed using the dot product of extracted features and the coefficient vectors. To get the probability of tag assignment given a token sequence, this term needs normalization by summation over all possible

tags represented by the term $Z(x[1], ..., x[N])$ as shown in Table 4.1(b). Though the features are extracted locally using the function *f*, the global normalization captures long distance relationships in the token sequences.

Training the CRF Model: Our objective is to spot event and location terms in tweets. Identifying locations in a tweet is challenging as location references are hard to recognize especially in the presence of non-standard abbreviations, spellings, and capitalization convention. A sample tweet with location and event annotations is shown in Figure 4.5. To address these challenges, we train the sequence model with the knowledge of locations from Open Street Maps (OSM) [58] for a specific city. OSM data is available for most of the cities around the world. Identifying event terms in tweets is challenging, especially given the open domain nature of city related events. Background knowledge consisting of domain vocabulary is obtained from 511.org, which provides a hierarchical classification of traffic related events. E.g., music event, sporting event, and road work that are categorized as *scheduled events* and accident, break down, and protests are categorized as *active events*. We generate training data automatically using the knowledge of locations and event terms using a dictionary based spotting. The training data may be cleaned before using it for training the CRF model. Cleaning refers to removing annotated tweets that have ambiguous references (Figure 4.3). Depending on the availability of resources, our city event annotation framework offers flexible manual control. Desired accuracy of spotting location and event terms would determine the extent to which the training data should be cleaned. Given the open domain nature of city events and robustness of our event extraction algorithms, it was not necessary to clean the training data. We compare our CRF model trained on this automatically created training data (without cleaning) with the baseline CRF model reference which is trained on manually created training data. Our approach shows promising results as evident by Figure 4.10 and Figure 4.11 based on the precision, recall, and F-measure metrics.

4.2.4.2 City Event Extraction

Using the named entities and event phrases extracted from tweets, we derive unique events in the city. There may be multiple references to the same event. Further, an event phrase may co-occur with multiple event types. For a reliable event extraction, we follow a systematic approach as outlined below (each component in the city event extraction box of Figure 4.6 is detailed here).

(1) Geohashing: We split the city into grids of a specified area using the geohashing algorithm[5]. These grids compartmentalize a city into various spatial regions. Different grids correspond to different levels of granularity. The spatial precision increases with the length of the string representing a location. We assign unique grid number to each grid. Figure 4.7 presents a geohashing example for San Francisco Bay Area and shows a tweet reported within the geohash. We associate a unique identifier to the location meta data of the tweet originating from that grid. Algorithm 1 transforms a raw tweet with timestamp (ts) and geo-location (lat and long) to feature vectors in the form of a quintuple $\langle n, e, t, d, l \rangle$ where, n, e, t, d, and l represents location terms, event terms, time of day, day of week, and geohash location respectively. The CRF model is trained on automatically generated training set (using dictionary based spotting).

(2) Event Aggregation: After careful consideration of event characteristics in a city, we make *three assumptions* to group messages with event terms and location annotations in a city: (a) *Spatial Coherence*: Events reported within a grid $g_i \in G$ (where G is a set of all grids in a city) in the same time interval are associated with the same event. (b) *Temporal Coherence*: Events reported within a time interval Δt (difference between end time and start time) in a grid g_i are associated with the same event. (c) *Thematic Coherence*: Events with similar entities reported within a grid g_i and time Δt are associated with the same event.

Algorithm 2 presents our approach to derive city traffic related events from the feature vectors generated by Algorithm 1. The input to the algorithm are the feature vectors generated within a

[5]http://wiki.xkcd.com/geohashing/The_Algorithm

Algorithm 1: Populating metadata for each tweet

Input: $tweet_{ts,lat,long}, CRF_{model}$

Output: $\langle n, e, t, d, l \rangle$

n := spotEntities($tweet_{ts,lat,long}, CRF_{model}$);

e := spotEventTerms($tweet_{ts,lat,long}, CRF_{model}$);

t := getTimeOfDay($tweet_{ts,lat,long}$);

d := getDayOfWeek($tweet_{ts,lat,long}$);

l := getGridNumberFromGeohash($tweet_{ts,lat,long}$);

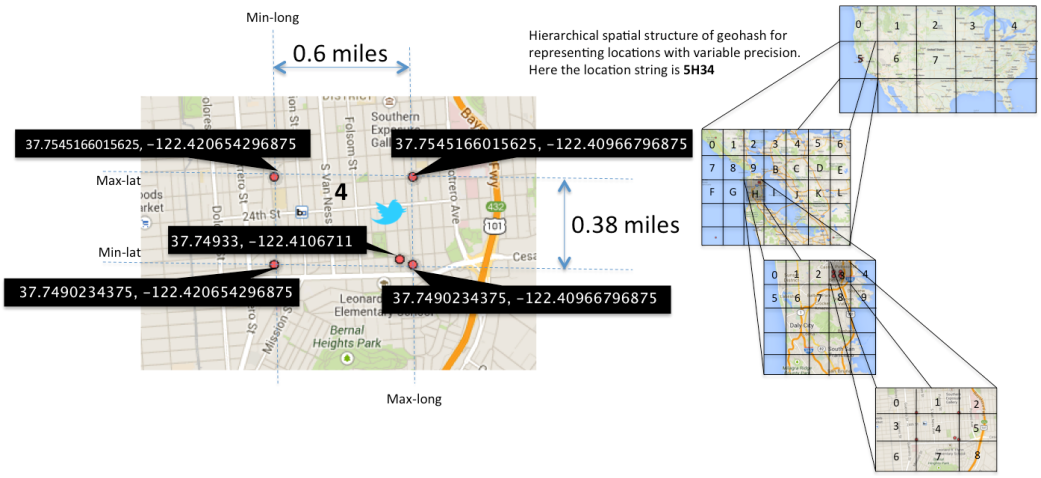
return $\langle n, e, t, d, l \rangle$;

time interval Δt. The algorithm utilize the set of grids (squares) in a city G generated using a geohashing implementation[6] and the event hierarchy from 511.org. Algorithm 2 has three steps. First, each feature vector $\langle n, e, t, d, l \rangle$ associated with individual tweet is assigned an event type based on the event hierarchy of 511.org. The event type is assigned based on the event term e in the feature vector $\langle n, e, t, d, l \rangle$. Since the CRF model is trained using the 511.org hierarchy, the event term e would correspond to a type in the hierarchy. Second, the feature vectors are grouped together based on space, time, and theme information. Finally, in the third step, each event type cluster (set of feature vectors) is processed for gleaning the start and the end time, location, and impact of the event. Each feature vector has a unique grid associated with it and each grid is assumed to have a unique event. Start time of the event is approximated using the time stamp associated with the first tweet (determined by timestamp) in the cluster. End time of the event is estimated using the timestamp associated with the last tweet (determined by timestamp) in the cluster. For estimating event location, we count the maximum number of occurrences of location l_{max} in the event type cluster.

(3) Impact Assessment: Events may have varying impact on the functioning of a city. City

[6]https://github.com/kungfoo/geohash-java

Figure 4.7: Spatial region bounded by a box which is part of the geohashing scheme to split a huge geographical area into smaller addressable units. A tweet posted within this box is shown.



authorities need to prioritize these events based on the severity level. For example, a pot hole on a major road can be more critical to fix than a pot hole on a smaller road that is used much less. City authorities have realized the importance of impact assessment of city events[7]. Unlike formal incident reports, tweets do not contain easily accessible/decipherable information. We approximate the seriousness of an event by the number of people reporting the event. Algorithm 2 captures this intuition for estimating the event impact.

[7]http://www.kaggle.com/c/see-click-predict-fix

Algorithm 2: Derive event descriptions from feature vectors generated by Algorithm 1

Input: Representation of tweet content using quintuple $\langle n, e, t, d, l\rangle_N$ where n represents location terms, e consists of the event terms, t represents the time of day, d represents the day of week, and l represents the geohash location, and N refers to the number of input quintuples, Δt representing the time step such as hour, day, or week used to step through starting time t_s and ending time t_e

Output: Event quintuple corresponding to a collection of tweets giving the aggregated type, location, start time, end time, and impact represented by $\langle \hat{e}_{i,type}, \hat{e}_{i,loc}, \hat{e}_{i,st}, \hat{e}_{i,et}, \hat{e}_{i,impact}\rangle_n$ where n represents the number of events

while $\exists\langle n, e, t, d, l\rangle$ *within a time step* Δt **do**

// Associate a type with each quintuple utilizing the 511.org event hierarchy

for *i := 1:N* **do**

$v_i := \langle n, e, t, d, l\rangle_i$;

type := 511.org hierarchy term associated with e ;

Assign event type *type* to v_i ;

end

// Create event type buckets with corresponding feature vectors

for *i := 1:N* **do**

Collect all the feature vectors v_i with the same *type* into an event type bucket $E[type_k]$ where k is the number of event types ;

end

// Filter cluster items based on grid information

for *i := 1:k* **do**

Find location with highest number of occurrence in $E[type_i]$ represented by l_{max} ;

Remove all the members of the set $E[type_i]$ whose location is not l_{max} ;

end

// Derive event metadata from event clusters

for *i = 1:k* **do**

$\hat{e}_{i,type} := type_i$;

$\hat{e}_{i,loc} := l_{max}$ associated with $E[type_i]$;

$\hat{e}_{i,impact}$:= number of items in the set $E[type_i]$;

$\hat{e}_{i,st}$:= smallest time stamp in the cluster $E[type_i]$;

$\hat{e}_{i,et}$:= largest time stamp in the cluster $E[type_i]$;

emit $\langle \hat{e}_{i,type}, \hat{e}_{i,loc}, \hat{e}_{i,st}, \hat{e}_{i,et}, \hat{e}_{i,impact}\rangle$;

end

end

Overall, we presented an approach for city event extraction in two steps: (a) *Annotation*: we use the CRF model trained using the automatically generated training data (using dictionary based spotting of OSM and 511.org entities) to determine city locations and event terms. (b) *Extraction*: Using the three key assumptions of spatial, temporal, and thematic coherence characterizing city events, we aggregate feature vectors to glean event meta data.

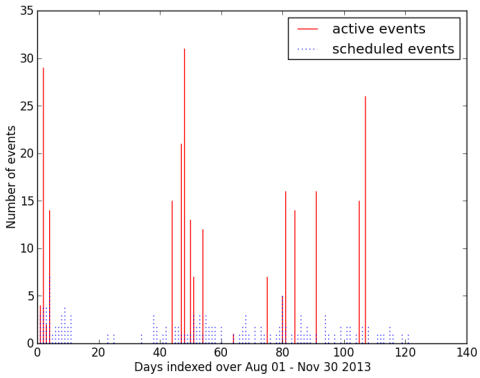
4.3 Evaluation: Traffic Event Extraction from Tweets

To evaluate our approach, we need to prepare training and test datasets, and train a CRF model for annotating tweets.

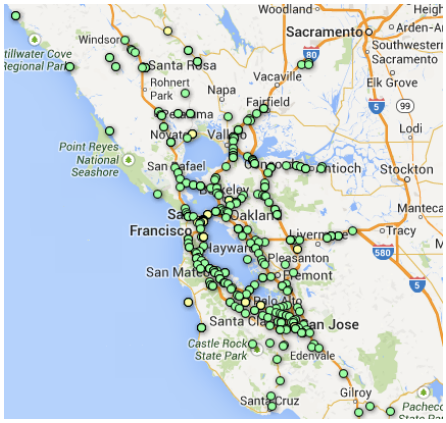
4.3.1 Dataset Description and Evaluation Metric

To make the evaluation tractable, we constrain our experiments to the domain of traffic related events. This was motivated by the availability of ground truth data from city authorities of San Francisco Bay Area[8]. The proposed approach is generic enough that it can be applied to any other domain for which the ground truth is available. We propose a novel approach to create massive training data with minimum manual intervention. We leverage two external sources in the work: (1) Open domain knowledge available for a city, specifically, vocabulary related to traffic from 511.org, and (2) OSM for city locations. For those domains not covered by the 511.org hierarchy, a vocabulary of event terms should be augmented. We have collected data from 511.org and twitter for a period of four months (Aug 2013 to Nov 2013). We utilized the Java Messaging Service (JMS) to receive the traffic data in the form of an XML stream from 511.org. For collecting twitter data, we used the twitter streaming API with location bounding box as San Francisco Bay Area. There are over 8 million tweets collected for this time period, augmented with 162 million sensor data points, 180 scheduled events, and 335 active events. The total dataset size is around 7 GB. Incident reports and

[8]http://511.org/developer-resources_traffic-data-feed.asp



(a)



(b)

Figure 4.8: (Generated from Google Fusion Tables) Spatio-temporal distribution of ground truth data consisting of Active and Scheduled events over four months obtained from 511.org

sensor data from 511.org may serve as the ground truth (though we use only incident reports in this work). Table 4.2 summarizes active and scheduled events along with various sub-types. Temporal distribution of events over the period of four months (Aug-Nov, 2013) is shown in Figure 4.8(a). There are more active events compared to scheduled events and the distribution is non-uniform (unpredictable). Scheduled events are distributed uniformly throughout four months. A spatial distribution of events is also presented in Figure 4.8(b). Traffic events are concentrated on major roads and central part of the city. Our objective of the evaluation is to quantify the extent to which our approach can recover traffic incidents from tweets. We compare our approach with a state-of-the-art baseline [122] using precision, recall, and F-measure along with confusion matrix.

4.3.2 Training Data Creation

We use a novel approach to create city specific training data for sequence labeling task by utilizing the domain knowledge of city locations and event vocabulary.

Table 4.2: Ground truth events collected from 511.org along with their number of occurrence between August 2013 and November 2013

Active Events

incident;truck-fire	2
special-events;festival	1
incident;emergency-maintenance	1
incident;accident-involving-a-motorcycle	1
obstructions;downed-power-lines	2
traffic-conditions;residual-delays	3
sporting-events;race-event	1
incident;disabled-semi-trailer	6
incident;accident	73
disasters;grass-fire	1
incident-response-status;police-department-activity	1
incident;injury-accident	13
obstructions;debris-on-roadway	3
incident;disabled-bus	1
incident;overturned-semi-trailer	1
visibility-air-quality;fog	1
incident;disabled-truck	3
disasters;fire	1
incident;multi-vehicle-accident	1
incident;spilled-load	1
sporting-events;baseball-game	3
incident;vehicle-on-fire	1
special-events;fair	9
roadwork;long-term-road-construction	7
incident;disabled-vehicle	9
incident;single-vehicle-accident	2
incident;road-construction	92
incident;spinout	1
obstructions;obstruction-on-roadway	38
winds;strong-winds	6
device-status;signal-problem	1
special-events;major-event	1
Total Number of Active Events	*311*

Schedule Events

race-event	5
fair	24
movie-filming	1
festival	5
long-term-road-construction	5
football-game	26
hockey-game	14
basketball-game	10
concert	25
race-eventmarathon	2
road-constructionpaving-operations	2
major-event	6
weekend-long-construction	1
soccer-game	1
concertfestival	1
baseball-game	42
Total Number of Scheduled Events	*170*

4.3.2.1 Data Preprocessing

We create training data with location name annotations utilizing locations from Open Street Maps (OSM) [58] and event term annotations using the hierarchical knowledge of traffic events from 511.org. There are two levels of filtering: (a) *Location based filtering* using the latitude-longitude of a bounding box around the city to filter tweets from the city, and (2) *Content based filtering* using location names on OSM and traffic related concepts on 511.org to filter tweets related to the domain of traffic events in the city. We propose a novel scalable solution for creating training data that utilizes available knowledge as a dictionary to annotate real-world data collected from twitter. We use the Aho-Corasick [3] string matching algorithm implemented by LingPipe [5] to perform annotation of locations/event terms in linear time. The Aho-Corasick algorithm utilizes the dictionary of locations and event terms to spot entities in twitter text. Annotated tweets containing location and event terms are then used as a training sample for building a CRF model.

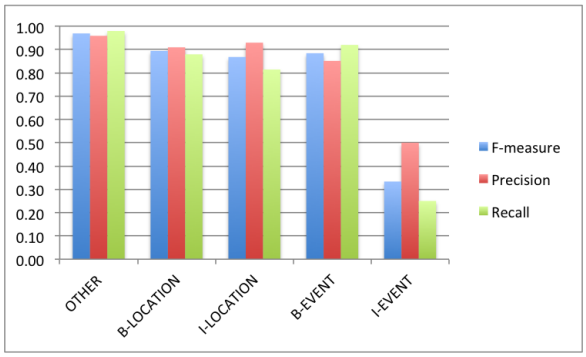
Table 4.3: Evaluation results of the dictionary based training data creation process using precision, recall, and F-measure

		Actual Labels					Total	Precision
		OTHER	B-LOCATION	I-LOCATION	B-EVENT	I-EVENT		
Dictionary Annotation	OTHER	4267	62	113	6	3	**4451**	0.96
	B-LOCATION	37	451	7	1	0	**496**	0.91
	I-LOCATION	39	0	525	0	1	**565**	0.93
	B-EVENT	12	0	0	80	2	**94**	0.85
	I-EVENT	2	0	0	0	3	**4**	0.50
Total		**4357**	**513**	**645**	**87**	**8**		
Recall		0.98	0.88	0.81	0.92	0.25		
F-measure		0.97	0.89	0.87	0.88	0.33		

4.3.2.2 Preprocessing Evaluation

To understand the quality of training data, we evaluate the automatic training data creation process. The evaluation is carried out by random sampling of annotated tweets. We select 500 random samples with 5,616 tags for evaluation. These samples were obtained from tweets generated during three months from Aug 2013 to Oct 2013. The results of evaluation are presented in Table 4.3. We define *precision* as the ratio of number of correctly classified instances (tags) to the total number of

Figure 4.9: Plot of Precision, Recall, and F-measure for the dictionary based training data creation process



instances. Specifically, *precision* is defined for each label by the ratio of the diagonal entry in Table 4.3 and the corresponding entries in the *Total* column in Table 4.3. The total number of instances is the sum of correctly classified instances, N_C (sum of diagonal elements in Table 4.3) and incorrectly classified instances, N_I (sum of non-diagonal elements in Table 4.3). Our annotation process exhibits high quality with precision of around 94%. With such an accuracy, human intervention can be minimized or even eliminated. Since precision alone cannot provide insights into the annotation process, we present our results in the form of a confusion matrix in Table 4.3. Recall that B- and I- refer to the beginning and intermediate tags respectively when there are multiple words in an entity name. *Location* and *event* are the two types of entities that contribute to event metadata. The *Other* tag is assigned to any other category of tokens. The loss of precision is due to the following challenges: (1) Subtle change in context results in varied interpretation of words, e.g., "Bed bath and beyond aha" can refer to a location or it is a casual remark not related to any location. This lack of context caused our annotator to mark this as location while in the evaluation we took a conservative approach of penalizing such annotations. (2) Intertwined space and time references cause loss of precision, e.g., "All them people from middle school and high school I don't even talk to them anymore just shows me a lot" in which the author is not really referring to any location but merely referring to temporal dimension of life. Our annotator cannot differentiate between such references. (3) Subtle difference in location and event references, e.g., Twin Peaks in "Twin

Peaks Summit" is labeled as location since Twin Peaks is actually a location. The word "Summit" makes the interpretation of the entire phrase as an event. Since our dictionary based annotation process is context-free, it cannot catch such subtle differences. All these limitations motivated us to move toward a tagger that can capture such dependencies between words. We utilized 8,074 annotated tweets as a training set for building a CRF model which addresses some of the limitations we described in this section.

4.3.3 Model Creation and Evaluation

We compare the CRF model created using our approach (with no manual intervention in creating the training data) with the baseline [122] which was trained on a manually crafted dataset. A quantitative comparison of the two approaches for the annotation task is presented here.

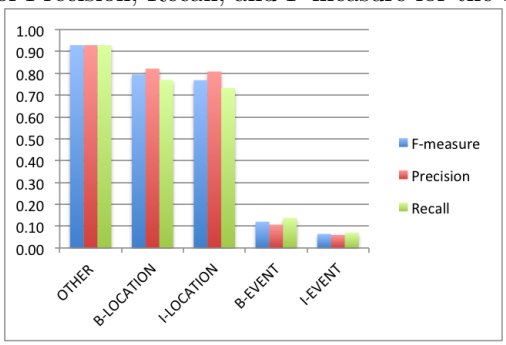
We used 8,074 annotated tweets to train a linear chain CRF model. The trained CRF model, created using 8,074 annotated tweets, is used in Algorithm 1 for annotating location and event tokens. We use the CRF implementation provided by LingPipe [5] for our experiments. We evaluate the tagging process on the data collected for the month of Nov 2013 (test data) which is not used in any of the previous experiments. This temporal separation of data is natural in the context of temporal streams such as microblogs. For scalability of our approach it is necessary to create training data in an autonomous manner. To explore this, we compare two scenarios. First, we evaluate the CRF model created by using the annotated data from the previous section as is. Evaluation is done with manual inspection in which location and event annotations are examined for correctness. Second, we annotate the microblog text using the baseline approach [122] and evaluate the quality of annotation. We carry out the two experiments on 500 randomly chosen tweets from Nov 2013. The performance of annotation is evaluated using a confusion matrix for a deeper insight along with the precision, recall, and F-measure scores.

The baseline model [122] is trained on a carefully annotated tweet corpus in three different categories of annotation. The first training dataset is for Part Of Speech (POS) tagged tweets.

Table 4.4: Evaluation of annotation based on precision, recall, and F-measure metrics for baseline

		Actual Labels					Total	Precision
		OTHER	B-LOCATION	I-LOCATION	B-EVENT	I-EVENT		
Baseline Annotation	OTHER	6015	95	139	192	34	**6475**	0.93
	B-LOCATION	68	327	2	1	0	**398**	0.82
	I-LOCATION	89	3	392	1	0	**485**	0.81
	B-EVENT	259	0	2	32	6	**299**	0.11
	I-EVENT	41	0	0	6	3	**50**	0.06
Total		**6472**	**425**	**535**	**232**	**43**		
Recall		0.93	0.77	0.73	0.24	0.07		
F-measure		0.93	0.79	0.77	0.12	0.06		

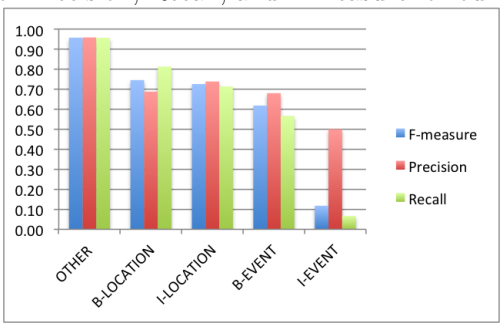
Figure 4.10: Plot of Precision, Recall, and F-measure for the baseline annotation



This data consists of tweets annotated with POS tags. Second training dataset consists of the tweet chunking information which has tags grouping the beginning and end of POS tags in a tweet, e.g., to capture multi-word nouns. Third training dataset consists of the named entities. These datasets[9] are annotated using the BIO notation. The baseline dataset is created meticulously by manual inspection of tweets. This is an arduous task given the volume of tweets and the challenges in understanding tweet content. Sometimes the lack of context is so serious that it can confound manual annotation. The precision of the baseline annotation task is shown in Table 4.4. The model suffers loss of precision mostly for the event term annotation justified by the lack of background knowledge of event terms. The baseline takes a high-recall low-precision approach so that the events can be ranked based on the rarity of events (done during the event aggregation phase). The precision, recall, and F-measure of the baseline approach is plotted in Figure 4.10. To understand the impact of event term annotation on the overall precision, we need to change the denominator term N_I, where N_I is the sum of all the non-diagonal entries which constitutes the mis-classified instances.

[9]https://github.com/aritter/twitter_nlp

Figure 4.11: Plot of Precision, Recall, and F-measure for our annotation process



Consider the first column of Table 4.4. Number of instances that belonged to 'other' category but were classified as B-EVENT is given by the fourth entry in the first column. If we retain this, since baseline is based on the high-recall philosophy, we are unnecessarily penalizing the baseline. We can pretend that the baseline did not provide us with these event tags and that it was tagged as other. This results in the modified precision for the baseline computed as

$$Precision_{baseline}^{ignore-event-annotation} = \frac{N_C}{N_C + N_I} = \frac{6769 + 259}{6769 + 259 + (938 - 259)} \approx 91\%$$

where, N_C, the number of correctly classified instances is given by the sum of all entries along the diagonal in Table 4.4. N_I is obtained by the sum of all the non-diagonal entries in Table 4.4.

Table 4.5: Evaluation of annotation based on precision, recall, and F-measure metrics for our approach

		Actual Labels					Total	Precision
		OTHER	B-LOCATION	I-LOCATION	B-EVENT	I-EVENT		
CRF model Annotation	OTHER	5741	69	125	36	20	**5991**	0.96
	B-LOCATION	123	313	17	2	0	**455**	0.69
	I-LOCATION	125	1	361	0	2	**489**	0.74
	B-EVENT	14	2	2	51	6	**75**	0.68
	I-EVENT	0	0	1	1	2	**4**	0.50
Total		**6003**	**385**	**506**	**90**	**30**		
Recall		0.96	0.81	0.71	0.57	0.07		
F-measure		0.96	0.75	0.73	0.62	0.12		

The evaluation of our approach is presented in the form of a confusion matrix in Table 4.5. The proposed approach essentially takes a dictionary of terms as input for creating a training data without any manual intervention. The dictionary used here are Open Street Maps and the event related knowledge from 511.org. Such a dictionary consists of the vocabulary used in referring to

Table 4.6: Normalization of tags with baseline tag and the corresponding normalizing tag

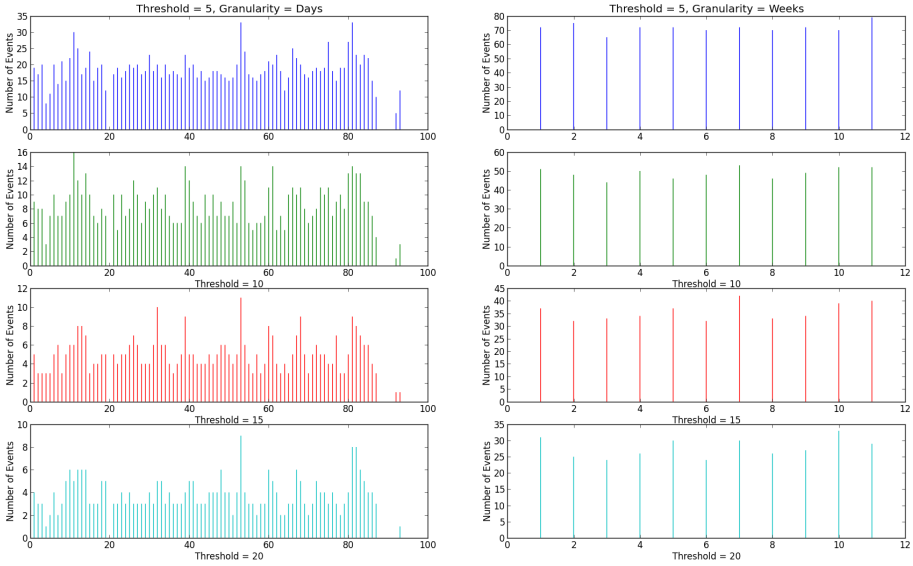
Baseline tags	*Normalizing tag*
B-facility, B-company, B-geo-loc	B-LOCATION
I-facility, I-company, I-geo-loc	I-LOCATION
B-event	B-EVENT
I-event	I-EVENT
B-other, I-other, B-person, I-person, B-tvshow, I-tvshow, B-sportsteam, I-sportsteam, B-movie, I-movie, B-product, I-product, B-musicartist, I-musicartist	OTHER

various concepts in the domain and such a vocabulary is readily available for most of the cities. We did not perform any cleaning of the dataset before training the CRF model. Precision, recall, and F-measure of our approach is shown in Figure 4.11. This is on a par with the precision of the baseline while reducing the tremendous effort involved for manually created training data. A city may have many other events of interest. We need a scalable approach that can leverage existing knowledge of the domain to feed the statistical NLP models (e.g., CRF) with automatically created training data. Our results show that this is indeed possible.

4.3.3.1 Principles for annotation of ground truth

We use the same ground truth to compare CRF model built on manual and automatically annotated corpus. The ground truth was created as follows: (1) *Normalization*: We perform tag normalization associated with locations and events by making the transformations as shown in Table 4.6. (2) *Minimality*: When we need to decide on including or excluding neighboring words within the location annotation, we look for the minimum words that provide a unique location hit on google maps. We stop including words that do not really contribute to location uniqueness, e.g., if *Bay Area Medical Academy San Francisco* appears in text, we believe that the first four words are enough to get a hit on google maps. (3) *Specificity*: Annotating specific locations at the expense of missing general locations is acceptable. We do this since we can infer the generic location from the specific location e.g., if *Bay Area Medical Academy* and *San Francisco* appear in a tweet, annotating the first location will allow us to infer the second location.

Figure 4.12: Sensivity of event extraction to thresholds presented for the time granularity of days and weeks: A consistent pattern in the number of extracted events is observed over various thresholds



4.3.4 Scalability Challenges

We started by exploring the use of existing tools for building the CRF models. MALLET [100] is a comprehensive tool for Natural Language Processing with a suit of machine learning libraries. The learning technique used is the Limited-memory BFGS (L-BFGS) does not scale for the dataset we have. LingPipe [5] provides a CRF library which utilizes Stochastic Gradient Descent (SGD) as the learning mechanism and scales well for our dataset size.

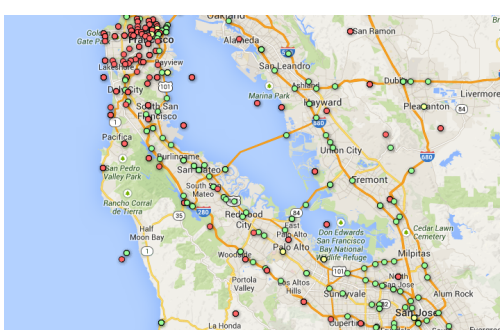
4.4 Evaluation of Event Extraction

An implementation of all the algorithms presented in this Chapter along with complete dataset is available as an Open Science Framework project[10]. We evaluate the effectiveness of the events extracted and its relationship to incident reports by investigating if they are corroborative, com-

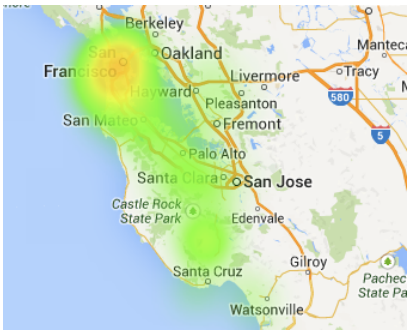
[10]https://osf.io/b4q2t/

plementary, or timely compared to the incident reports from 511.org. We believe that microblog alerts such as tweets are related to conventional sources such as news and incident reports by city authorities and sensor data in several ways. We call the event extracted from twitter as *corroborative* to the event from 511.org if they are reporting exactly the same event. We call the extracted event to be *complementary* if it provides additional information to 511.org events, e.g., extracted event may be traffic jam that further adds to the construction event from 511.org. The extracted event (complementary or corroborative) is called *timely* if it precedes the event reported on 511.org. They may provide additional information and may even help us explain some observations reported on conventional sources. For example, we extracted 'traffic' event from a textual stream and a 'baseball game' observation as reported from 511.org, and both these events have same space and time extent. Traffic information is complementary to the baseball game. If we extract baseball game event from textual stream, then the event will be corroborative (one supporting the other). If we extract any of the two events (traffic or baseball game in this example) before the incident report from 511.org then the extracted event is timely. We use these "characteristics" to manually verify each extracted event. The bottom-up nature of citizen sensing has both positive and negative implications. A positive implication is that such near real-time reports may surpass the conventional sources in terms of timeliness. A negative implication is that they can be contradictory or misleading.

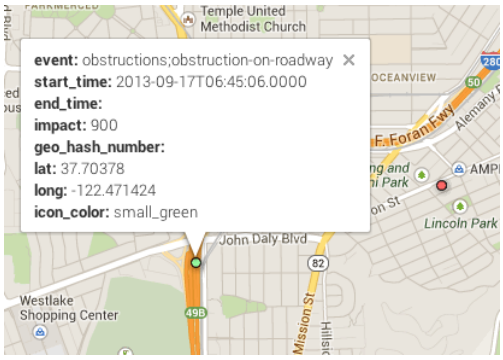
For better understanding of the distribution of extracted events over time, we vary the granularity of time slices (days and weeks) and the threshold of minimum concurring tuples that constitute an event (5, 10, 15, and 20). In other words, our confidence on event occurrence depends on the cluster size. We study the variation of number of event tuples emitted based on a threshold of 5, 10, 15, and 20. We perform this study over days and weeks by varying granularity of time as shown in Figure 4.12. We chose days and weeks as time granularity and exclude months as it may not provide any additional insight. The x-axis is the time granularity and the y-axis is the number of events. An important thing to note is that the variation of threshold from 5 to 20 has not created any major change in the distribution of events over time. When threshold is low, there are lot more events



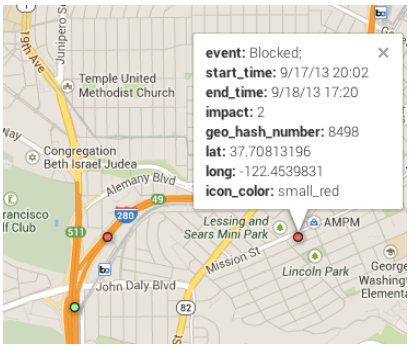
(a) Events from twitter and 511.org



(b) Heatmap of city traffic events



(c) Event from 511.org reporting obstruction



(d) Event from twitter reporting blockage

Figure 4.13: (Generated from Google Fusion Tables) Distribution of city events that were extracted from tweets along with the scheduled and active events from 511.org

reported per day. From the first row of Figure 4.12, we see that an average of 20 events are detected per day and 70 events per week. This is still small compared to the total number of tweets that are generated every day (which is in the thousands). There is a 50% drop in the number of events over days for every 5 unit increase in the threshold. For a threshold of 20, there are around 4 traffic events per day.

A distribution of extracted events from twitter and 511.org events over the city of San Francisco are shown in Figure 4.13. Recall that the data was collected from Aug 2013 to Nov 2013. Our approach has been complementary to the standard sources of events in a city as testified by the red dots (extracted event) in areas where green and yellow dots (511.org events) are sparse. Green dots indicate active events (e.g., accidents, breakdowns, blocked roads) while the yellow dots indicate

scheduled events (e.g., baseball game, concert, maintenance). Also, the extracted events appear on highways or near major roads. Further, the events are concentrated in the city center where we expect a lot of activity as summarized by the heatmap.

Figure 4.13(c) has an active event reported by 511.org and Figure 4.13(d) is the city event extracted from twitter. The 511.org event is related to obstruction on roadway reported at 2013-09-17 06:45. The extracted event provides insight that the event has lasted from 2013-09-17 20:02 to 2013-09-18 17:20 beyond the event occurrence time. Such complementary information will be useful for understanding the impact of an event.

We summarize the result of evaluation for 1,042 city events extracted from all the tweets collected over four months by considering 26 of them for clarity in this section. The event extraction was done using a threshold of 10 as we wanted to strike a balance between the number of events and the manual effort involved in examining the results. The extracted city events and a comparison to ground truth (corresponding entries in Table 4.8), in terms of whether they are complementary (CP), corroborative (C), or timely (T) events, are presented in the Table 4.7. Each tuple in Table 4.7 represents an event and the subscript of the tuple refers to the relationship of the event to the corresponding ground truth in Table 4.8 containing scheduled (s) and active (a) events. Note that there are 26 lines in both the tables with line to line correspondence. Table 4.9 lists all the event-types for the extracted events. Events 1, 3, 4, 7, 8, 12, 15, 16, 20, 21, 22, 23, 25, and 26 provide a complementary view of bad traffic conditions during events of road constructions, baseball games, football games, and hockey games. Events 2, 5, 6, 9, 10, 14, 17, 18, and 19 are corroborative, reporting events that are consistent with events from 511.org. Events 5, 10, 13, 14, 15, 18, 19, 22-25 are reported before the events from incident reports of 511.org making twitter a timely source of information for city authorities to react. The dataset[11] used in this evaluation, consisting of twitter data and 511.org data, are available as an Open Science Framework project[12] for the research community.

[11] http://harp.cs.wright.edu/cityevents/

[12] https://osf.io/b4q2t/

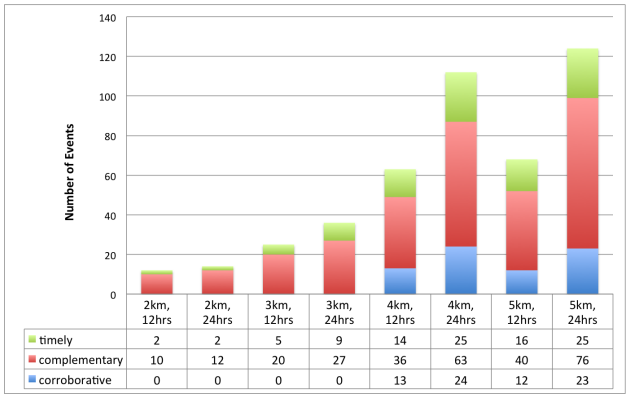
Table 4.7: Events extracted from textual stream compared with ground truth of 511.org categorized as C = corroborative, CP = complementary, and T = timely

1	⟨ traffic, [37.642231, -122.426173], 2013-07-31 19:48:33, 2013-08-01 19:02:46, 31⟩$_{CP}$
2	⟨festival, [37.35676, -122.117852], 2013-08-04 19:14:34, 2013-08-05 03:21:51, 30⟩$_{C}$
3	⟨traffic, [37.77752898, -122.4034819], 2013-08-25 19:35:26, 2013-08-26 18:06:21, 14⟩$_{CP}$
4	⟨traffic, [37.78151103, -122.4189619], 2013-09-03 20:03:19, 2013-09-04 18:53:31, 41⟩$_{CP}$
5	⟨concert, [37.35676, -122.117852], 2013-09-11 21:09:20, 2013-09-12 18:49:57, 29⟩$_{C,T}$
6	⟨football game, [37.39611, -121.931096], 2013-09-13 19:20:54, 2013-09-14 18:58:32, 14⟩$_{C}$
7	⟨festival, [37.35676, -122.117852], 2013-09-15 19:13:52, 2013-09-16 08:24:00, 35⟩$_{CP}$
8	⟨blocked, [37.77322896, -122.4254819], 2013-09-18 20:00:41, 2013-09-19 19:03:12, 18⟩$_{CP}$
9	⟨concert, [37.561391, -122.096567], 2013-10-07 19:29:54, 2013-10-08 18:40:41, 22⟩$_{C}$
10	⟨concert, [37.35676, -122.117852], 2013-10-09 19:21:01, 2013-10-10 18:22:25, 28⟩$_{C,T}$
11	⟨accident, [37.517208, -121.948119], 2013-10-10 21:57:43, 2013-10-11 18:12:01, 14⟩$_{CP}$
12	⟨traffic, [37.707621, -122.340281], 2013-10-13 19:27:10, 2013-10-14 16:45:49, 27⟩$_{CP}$
13	⟨fog, [37.77578298, -122.5136819], 2013-10-18 19:45:36, 2013-10-19 15:23:28, 14⟩$_{C,T}$
14	⟨festival, [37.604053, -122.472817], 2013-10-18 21:43:32, 2013-10-19 18:37:36, 30⟩$_{C,T}$
15	⟨traffic, [37.39611, -121.931096], 2013-10-18 19:15:36, 2013-10-19 15:40:18, 11⟩$_{CP,T}$
16	⟨traffic, [37.77994998, -122.4591199], 2013-10-18 19:16:12, 2013-10-19 18:32:52, 46⟩$_{CP}$
17	⟨fog, [37.561391, -122.096567], 2013-10-21 20:01:34, 2013-10-22 18:37:42, 45⟩$_{C}$
18	⟨concert, [37.329895, -122.065265], 2013-10-21 19:32:02, 2013-10-22 19:07:38, 43⟩$_{C,T}$
19	⟨accident, [37.77322896, -122.4254819], 2013-10-22 19:23:43, 2013-10-23 18:08:55, 14⟩$_{C,T}$
20	⟨traffic, [37.77322896, -122.4254819], 2013-10-30 20:22:54, 2013-10-31 11:14:44, 21⟩$_{CP}$
21	⟨traffic, [37.79262801, -122.4063839], 2013-11-03 13:03:42, 2013-11-03 21:43:06, 11⟩$_{CP}$
22	⟨traffic, [37.474743, -122.303362], 2013-11-13 00:49:38, 22013-11-13 22:57:06, 27⟩$_{CP,T}$
23	⟨traffic, [37.200495, -122.202653], 2013-11-14 01:15:10, 2013-11-14 23:29:47, 24⟩$_{CP,T}$
24	⟨tornado, [37.77502896, -122.4384818], 2013-11-17 01:38:36, 2013-11-17 19:03:57, 11⟩$_{CP,T}$
25	⟨traffic, [37.76405301, -122.4066841], 2013-11-22 02:51:30, 2013-11-22 22:12:16, 10⟩$_{CP,T}$
26	⟨traffic, [37.39611, -121.931096], 2013-11-27 00:40:34, 2013-11-28 00:06:56, 89⟩$_{CP,T}$

Table 4.8: Incident reports from 511.org that corresponds to the extracted events in Table 4.7 with subscript a = active events, s = scheduled events

1	⟨incident;road-construction, [37.628892, -122.41652], 2013-07-31T09:19:46.0000, 1800⟩$_a$
2	⟨fair, [38.433036, -122.703], 2013-08-04T10:00:00.0000, 2013-08-04T23:00:00.0000⟩$_s$
3	⟨football-game, [37.715272, -122.387296], 2013-08-25T13:00:00.0000, 2013-08-25T21:00:00.0000⟩$_s$
4	⟨baseball-game, [37.778752, -122.390288], 2013-09-03T18:15:00.0000, 2013-09-09T23:00:00.0000⟩$_s$
5	⟨concert, [37.423516, -122.07812], 2013-09-14T09:00:00.0000, 2013-09-14T23:00:00.0000⟩$_s$
6	⟨football-game, [37.87112, -122.251824], 2013-09-14T11:59:00.0000, 2013-09-14T20:00:00.0000⟩$_s$
7	⟨concert, [37.423516, -122.07812], 2013-09-15T10:00:00.0000, 2013-09-15T23:00:00.0000⟩$_s$
8	⟨incident;accident, [37.768712, -122.407712], 2013-09-17T17:53:53.0000, 900⟩$_a$
9	⟨concert, [37.332192, -121.900544], 2013-10-07T18:30:00.0000, 2013-10-07T23:00:00.0000⟩$_s$
10	⟨concert, [37.423516, -122.07812], 2013-10-12T18:00:00.0000, 2013-10-12T23:00:00.0000⟩$_s$
11	⟨baseball-game, [37.750956, -122.202232], 2013-10-10T16:00:00.0000, 2013-10-10T21:15:00.0000⟩$_s$
12	⟨football-game, [37.715272, -122.387296], 2013-10-13T09:30:00.0000, 2013-10-13T18:00:00.0000⟩$_s$
13	⟨visibility-air-quality;fog, [37.810832, -122.477416], 2013-10-19T22:55:47.0000, 1800⟩$_a$, ⟨visibility-air-quality;fog, [37.818596, -122.478584], 2013-10-19T22:57:05.0000, 1800⟩$_s$
14	⟨festival, [37.43372, -122.468288], 2013-10-19T08:00:00.0000, 2013-10-19T18:00:00.0000⟩$_s$
15	⟨incident;road-construction, [37.395324, -121.873672], 2013-10-19T22:16:50.0000, 1800⟩$_a$
16	⟨visibility-air-quality;fog, [37.810832, -122.477416], 2013-10-19T22:55:47.0000, 1800⟩$_a$, ⟨visibility-air-quality;fog,[37.818596 -122.478584], 2013-10-19T22:57:05.0000, 1800⟩$_a$
17	⟨visibility-air-quality;fog, [37.5402, -122.06688], 2013-10-20T08:00:30.0000, 1800⟩$_a$
18	⟨concert, [37.332192, -121.900544], 2013-10-18T18:30:00.0000, 2013-10-18T23:00:00.0000⟩$_s$
19	⟨incident;accident, [37.749184, -122.4038], 2013-10-23T08:32:18.0000, 1800⟩$_a$
20	⟨incident;road-construction, [37.788776, -122.387808], 2013-10-30T00:00:57.0000, 28800⟩$_a$
21	⟨football-game, [37.750956, -122.202232], 2013-11-03T09:00:00.0000, 2013-11-03T17:30:00.0000⟩$_s$
22	⟨incident;road-construction, [37.324488, -122.399984], 2013-11-13T07:05:06.0000, 28800⟩$_a$
23	⟨incident;road-construction, [37.258392, -122.122008], 2013-11-15T09:54:44.0000, 28800⟩$_a$
24	⟨winds;strong-winds, [37.779968, -122.398416], 2013-11-21T20:36:36.0000, 14400⟩$_a$
25	⟨incident;disabled-semi-trailer, [37.810656, -122.364336], 2013-11-22T08:44:45.0000, 1800⟩$_a$
26	⟨hockey-game, [37.332192, -121.900544], 2013-11-27T18:30:00.0000, 2013-11-27T23:00:00.0000⟩$_s$

Figure 4.14: Distribution of corroborative, complementary, and timely events across all the eight sets of event pairs



	2km, 12hrs	2km, 24hrs	3km, 12hrs	3km, 24hrs	4km, 12hrs	4km, 24hrs	5km, 12hrs	5km, 24hrs
timely	2	2	5	9	14	25	16	25
complementary	10	12	20	27	36	63	40	76
corroborative	0	0	0	0	13	24	12	23

4.4.1 Global Evaluation

We extend the evaluation to include all the 1,042 city events extracted from twitter and present a detailed evaluation by comparing them with all the 481 events from 511.org. Our evaluation strategy involves chunking the events based on location and time. We find out event pairs $\langle e_t, e_{511} \rangle$ that coexist within a radius of 2, 3, 4, and 5 kilometers where e_t is the event extracted from twitter and e_{511} is the event from 511.org. For each radius value, we find event pairs that coexist within the 12 hours and 24 hours window. Thus we have eight sets containing event pairs $\langle e_t, e_{511} \rangle$ each of which is evaluated for being complementary, corroborative, or timely. Figure 4.14 presents a distribution of extracted events from twitter as complementary, corroborative, or timely when compared to events from 511.org. The evaluation over all the eight sets containing event pairs is summarized. Although we have extracted many (1,042) city traffic events, around 40% of them (454) co-existed (based on location and time constraints stated above) with ground truth data. Our approach has potential to discover lot more city traffic events unreported on 511.org but we did not have the ground truth for verification.

Entity identification techniques such as sequence labeling are helpful in deciphering microblogs. The training data creation process that leveraged knowledge base of locations and event terms generated good quality training data. As such, the CRF model trained on this data performed on

Table 4.9: Types of events extracted from textual stream (originally from the 511.org hierarchy of traffic related events)

accident, blocked, left lane blocked, right lane blocked, baseball game, circus, cleared, concert, construction, crime, crowded, delay, dew, festival, fog, football game, frost, hurricane, incident, marathon, olympics, parade, performing arts, protest, rain, road construction, shooting, showers, snow, soccer game, toll plaza, tornado, tournament, traffic, weather

a par with a CRF model trained over manually created dataset. Furthermore, city events are open domain, so we need automated ways of creating training data for developing annotation models. The extracted events proved to be complementary, corroborative, or timely compared to the incident reports (from 511.org). As such, microblogs can serve as valuable enhancement to 511.org for analyzing and understanding road traffic.

Scalable training of sequence labeling models will be required for utilizing the automatically created training data. For better impact assessment, considering time of day, day of week, type of incident, etc., may be significant. Event extraction will help us know events, but to reveal valuable insights, we need to understand the relationships between various events. Additionally, we believe that declarative knowledge such as domain ontology or commonsense knowledge such as ConceptNet[13] would provide valuable support for understanding relationships between various events.

4.5 Traffic Event Extraction from SMS Messages

Cities around the world want to increase the adoption of public transportation modes such as metros and buses to reduce congestion and consequent pollution. A common obstacle for citizens in switching to public transportation is the lack of information about available choices when they need to travel. Although schedules of individual modes like bus or metro may be available as paper pamphlets, or digital files on websites, they do not give an integrated view of the complete services possible when a citizen actually wants to travel. Furthermore, the situation on the roads evolve and

[13]http://conceptnet5.media.mit.edu/

this demands timeliness of public transportation information. We want to tackle this problem in the context of cities of developing countries like India, which lacks basic instrumentation to track road conditions or vehicle location.

Our solution extends a public transportation recommender, working only with static schedule information, to utilize SMS messages about road conditions sent by city authorities. Our solution consists of: (a) extracting events from traffic alert messages, (b) reasoning about traffic delays from extracted events by qualitatively deciding what stops (locations) will be affected, and quantitatively estimating the lower bound on the probability of having a certain delay at those locations in the city, and (c) utilizing the delay estimates for route recommendation. We use publicly available traffic related SMS messages from Delhi, India, to evaluate our approach and show its promise. Our solution provides dynamic updates for transport network in cities with low investment and quick time to realization.

However, a common problem for citizens in adopting public transportation is the lack of information about available choices. Although schedules of individual modes may be available as paper pamphlets or digital files (e.g. pdf files, web pages), they are usually unavailable when needed, do not give an integrated view of the services possible in the city, and are not amenable to direct analysis. Furthermore, the situation on the roads evolve and this demands timeliness of public transport information. Decision support tools that can help commuters make journey decisions (journey planners) under such situations play a crucial role in promoting public transportation.

There have been many journey planners proposed in the literature and a few are available operationally. However, they rely on instrumentation on vehicles (e.g., GPS on buses, sensors on smart phones) to provide their recommendations. But, most of the cities around the world do not have such instrumentation in place - the time and investment needed to have sensors is long drawn while their citizens just have low-cost phones. They want journey planners which can work with current data and improve as better sensing becomes available.

Recently, a journey planner was proposed that used readily available schedule data from individ-

ual operators and processed them to establish the multi-modal service baseline [57]. However, since it worked only with static data, the recommendations could get stale from day to day or even time to time.

To get dynamic updates, we rely on a trend seen in many parts of the world where cities have an authorized, city-initiated, notification service in place to alert commuters about road conditions. For example, ten[14] cities in India including New Delhi [43], provide textual updates through city authorities as Short Message Service (SMS) to the subscribers. The city may themselves get the information from any existing sensor, their field employees, or community; all the information sent is verified. Hence, this becomes a reliable data source whose language and content are consistent. Assuming such a service is in place, we propose a journey planner that can (a) detect and classify events from textual notifications, (b) determine its impact on the multi-modal public transportation network, and (c) use it to rank travel choices least impacted. We develop such a system and show its feasibility in New Delhi, India, where no other similar capability exists. Our contribution include:

- feasibility study of using SMS updates for extracting traffic related events,
- estimation of the traffic disruption caused by event types at various locations in a city,
- estimation of (lower bound on) delays on public route networks based on traffic related events, and
- implementation of a live prototype for providing adaptive route recommendation for the city of Delhi.

In the rest of the Chapter, we first preview our work and then provide background followed by our approach. Next, we discuss an evaluation on real world data, put our approach in the context of related work and conclude.

[14]New Delhi (Delhi), Chennai, Bangalore, Pune, Mumbai, Chandigarh, Gurgaon, Nagpur, Kolkatta, Hyderabad.

SIno.	SMS messages
1	Traffic movement is slow from Dhaula kuan to Moolchand due to break down of a DTC low floor bus at the foot of Raj Nagar fly-over.msg@9.55am,200712.
2	Traffic is moving in one lane only on Burari road due to MCD work in front of Delhi Jal Board office.msg@10.46am,230612.
3	Due to construction work by DJB at Subhash nagar chowk (Tilak nagar towards Subhash nagar) half of the road is coverd by DJB therefore Traffic will remain heavy.msg at 08:56 p.m.,07/07/2012.
4	Traffic is moving slow at G.T.K Road from Mukarba chowk to Azadpur due to work by DJB.Kindly avoid this road.Use Mukund-pur to Azadpur road.msg@11.42am,180612.
5	Water logging on Lala Lajpat Rai Marg at North Foot & South foot of Defence Colony flyover, North foot of Moolchand flyover, left slip road from Ring Road to Lala Lajpat Rai Marg, oppo-site PS Defence colony Ring Road & opposite South Ex-2, Ring Road.msg7.26,130712.
6	The door of one of the compartment full of the stone of Maalgari at going towards Iron Bridge has open and stone have spilled on the road.msg@05.55pm260612

Table 4.10: A sample of traffic updates sent by Delhi Traffic Police in July 2012.

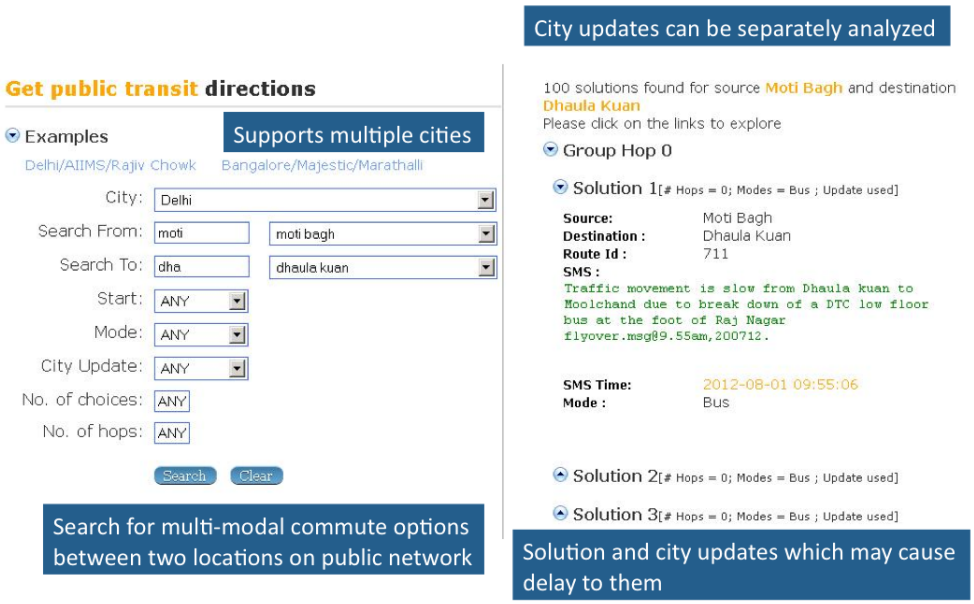


Figure 4.15: Snapshot of the journey recommender with dynamic updates.

4.6 Preliminaries

This section describes the real world data set used, event representation and its semantics, along with some concrete examples.

4.6.1 Datasets and Notation

New Delhi is the capital city of India as well as part of the bigger provincial state called Delhi. It has many public transportation services including a metro service run by Delhi Metro Rail Corporation (DMRC), a rail service by Indian Railways, bus services by the main public operator Delhi Transport Corporation (DTC) as well as public-private and private bus services by many other organizations. In the absence of any effective multi-modal transportation authority, the schedules of existing services are not available in a uniform format.

We will use the route dataset created by authors in [57] from schedules of bus services by DTC and metro by DMRC. Borrowing GTFS[55] terminology, a public transportation route is expressed as a 5-tuple, $\langle S, R, T, ST, F \rangle$: Stop ($S$), Route ($R$), Trip ($T$), Stop Times ($ST$), and Frequency ($F$). Figure 4.16 shows a sample.

Along with routes, we also need to know geographic details, e.g., of roads, on which transportation services are available. We use OpenStreetMap[113] that provides free geographic data and maps for most of the cities worldwide. We use it to associate transit stops (bus stops, metro stations) with locations on the city map.

Table 4.10 has SMS messages conveying road conditions, traffic movement, obstructions, and weather from SMSGupShup. A sample route recommendation incorporating dynamic updates is shown in Figure 4.15.

Agency

agency_id, agency_name, agency_url,
agency_timezone, agency_phone, agency_lang
DTC, Delhi Transport Corporation, http://www.dtc.com,
GMT + 530, 01123232433, en

Stop

stop_id, stop_code, stop_name
345, , Dhaula Kuan

Route

route_id, route_short_name, route_long_name, route_desc, route_type
R102, 711, 711, 711, 3
R103, Line 1,Line 1,Line 1,2

Trip

route_id, service_id, trip_id, direction_id
R102, A, 0A_R102, 0
R102, A, 1A_R102, 1

Stop Times

trip_id, arrival_time, departure_time, stop_id, stop_sequence
0A_R102, 0:0:00, 0:1:00, 345, 0
0A_R102, 0:6:00, 0:7:00, 412, 1

Frequency

trip_id, start_time, end_time, headway_secs
0A_R102, 7:25:00, 7:50:00, 5

Figure 4.16: A sample of multi-modal services made available in New Delhi (India) in GTFS format by authors.

4.6.2 Event Representation

Each event e^i is represented using a 6-tuple model $\langle type(e^i_{type}), description(e^i_{description}), location(e^i_{loc_{start}}, e^i_{loc_{end}}, e^i_{loc_{on}}), time(e^i_{time})\rangle$. The event type is an abstraction over collections of events following a common structure extracted from alerts. For example, break down of a Heavy Transport Vehicle (HTV) or break down of a car can be categorized as event type *BreakDown*. Event description can be details of the event (e.g., original message in case of traffic alerts may be event description). Event location has some nuances such as start location, end location, and on location. Event time plays an important role in assessing impact on public transport schedule. From the running example (the first message in Table 4.10), $\langle e^i_{type} = BreakDown, e^i_{loc_{start}} = DhaulaKuan, e^i_{loc_{end}} = Moolchand, e^i_{loc_{on}} = RajNagarFlyover, e^i_{time} = 20July, 2012, 9:55am\rangle$

4.7 Solution Components

4.7.1 Event Extraction

We extract events from update messages by specifying common patterns observed in the data we collected for two years for eleven cities in India. The current techniques uses regular expressions which provide precision of up to 96% and recall of up to 92%.

4.7.2 Reasoning Over Traffic Events

There are two logical steps for finding stops that are impacted by a traffic alert. The first step is to find “what stops are affected?" and the second step is to answer the question“by how much?"

Qualitative reasoning involves what stops are affected by a traffic event. The event locations extracted in the event extraction step ($e^i_{loc_{start}}, e^i_{loc_{end}}, e^i_{loc_{on}}$) are matched with a database of stop names for a city. The stop name matching is not limited to exact name match but we can use lat-long information obtained using OSM (Open Street Maps).

Quantitative reasoning involves finding the degree of impact on each stop due to a traffic event.

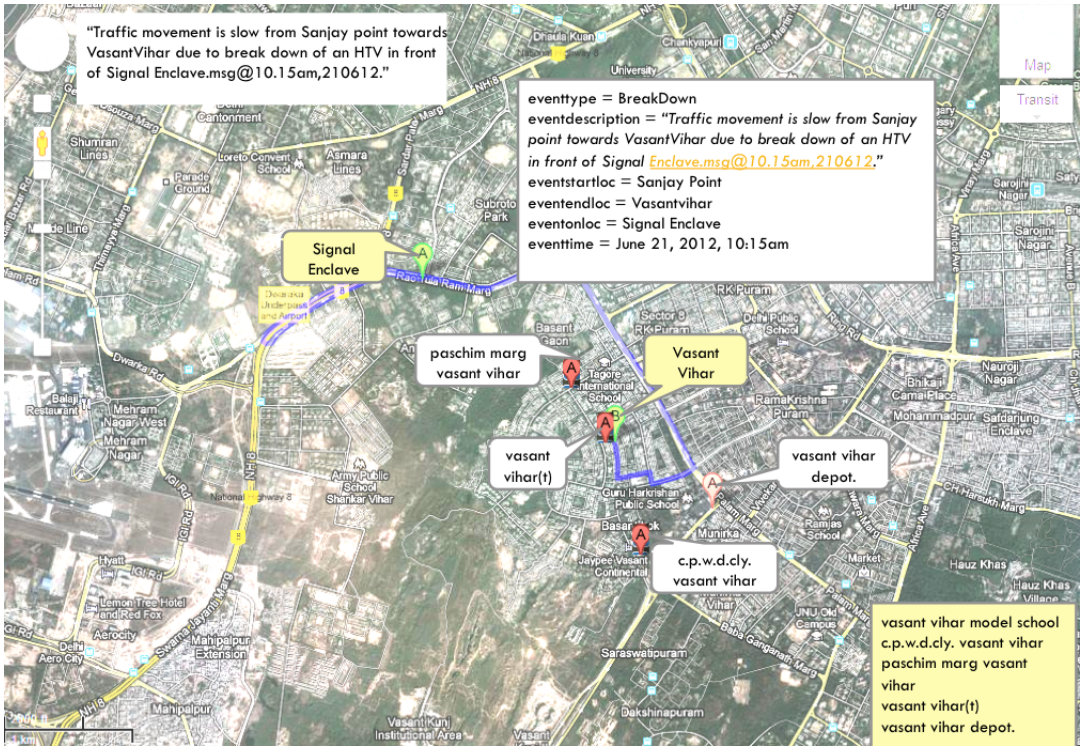


Figure 4.17: Illustration of notification, event extraction and its geographical extent.

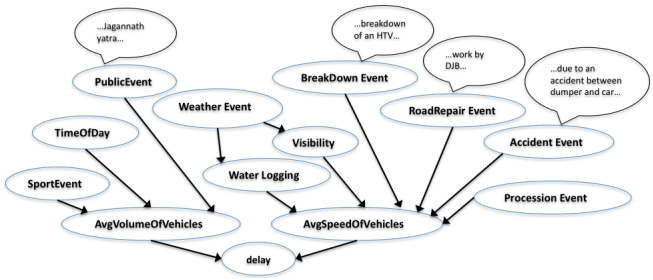


Figure 4.18: Bayesian Network for the domain of traffic along with sample instances of event types

We estimate the lower bound on the probability of delay ($\hat{P}_{delay}$) at a location based on event types reported at that location. This estimation is the lower bound on the delay probability (actual road conditions may be worse leading to higher than estimated delay) due to: (1) incomplete reporting of events, and (2) unknown interactions between different events.

Estimating Delay Probability: To answer queries on delay probabilities, we will consider questions such as: Will event type influence the delay? Will event types uniformly affect delays at different locations? Do locations have prior disposition for event types? We will reconsider these questions in the evaluation section.

An event has certain characteristic properties about how they impact traffic on a road. For example, breakdown of a car on a road may require a call to a mechanic to repair it or tow the car to the workshop. Calling the mechanic and getting the car repaired may lead to a delay of around 4 hours. In this instance, we were able to estimate the delay but it may not be feasible to do such a detailed analysis for all possible events.

This creates the need for a model which can learn the inherent event properties given partial domain knowledge (random variables and connection between them). Our objective is to determine the probability of having a commute delay for a link. For doing this, we need to (a) deal with incomplete, imprecise, and heterogeneous observations, (b) allow specification of domain knowledge for reasoning, (c) use data from the domain to validate and parameterize the domain knowledge, and (d) incorporate historical knowledge in the reasoning process.

We formulate the problem of finding the delay probability associated with a stop (location) using a Bayesian network [75], which combines domain knowledge of traffic and historical observations to reason over current observations of traffic.

Figure 4.18 is an example Bayesian network for the traffic domain along with some SMS observations from SMSGupShup (an SMS subscription service for traffic alerts). Each node in the network represents a random variable related to the traffic domain and each link imposes a structure among these random variables. All the random variables ending with “event" are the event types we have considered in the analysis.

We have sparse observations about the events in the form of messages, therefore the predicted probability of delays may not be accurate. Due to sparse observations, the priors play an important role in determining the accuracy of the reasoning process till we accumulate more observations. One way of initializing priors is by using historical traffic related events across different locations in a city. For instance, if the traffic accidents are found in the cities 20% of the time, then we say P(Accident) = 20%. The priors are initialized and personalized for each location in a city. We initialize the traffic event priors based on traffic alerts collected for two years from Delhi.

Assumption: We assume that each event has an independent effect on the delay. For instance, the event type *Accident* and *Procession* will have an independent effect on the delay. This is a reasonable assumption partly due to the unavailability of data. However, given data about multiple events and delays, we can still use the same framework to capture the collective effect of events on the delay probability estimation.

To assess the probability of having a commute delay at a stop, we associate a random variable $delay_{s_i}$ with all the stops in a route network represented by a set of delays, $\mathrm{D} = \{delay_{s_i}, \forall\ \mathrm{s}_i \in \mathrm{S}\}$. $delay_{s_i}$ at stop s_i is influenced by various traffic events at node s_i, $\mathrm{E}_{s_i} = \{e^1, e^2, \ldots, e^n\}$ where e^j is an event type. The probability of having a delay at a stop depends on the (1) events at the stop, and (2) prior probability of having a delay at the stop. The probability of having a delay at a stop given events observed at the stop in terms of the likelihood, and prior is given by $\mathrm{P}(delay_{s_i} \mid \mathrm{E}_{s_i})$

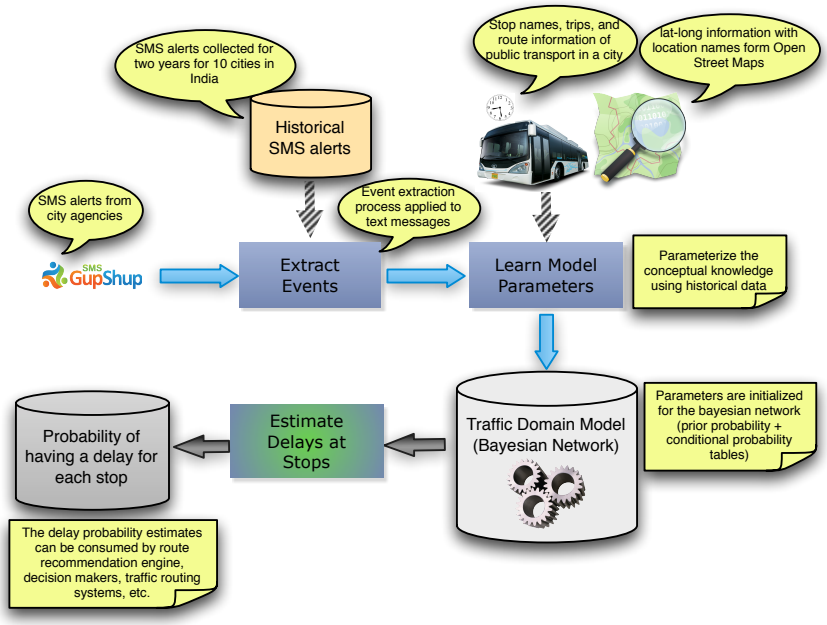


Figure 4.19: System architecture with event extraction and reasoning components

$= \mathrm{P}(\mathrm{E}_{s_i} \mid delay_{s_i})\ \mathrm{P}(delay_{s_i})\ /\ \mathrm{P}(\mathrm{E}_{s_i})$.

We illustrate parameterization of the Bayesian network for a place named Dhaula Kuan for clarity. We need to estimate the lower bound on the probability of having a delay at Dhaula Kuan. We noticed that accidents and breakdowns are the events that have major influence on delay at Dhaula Kuan using the traffic alerts for Delhi collected over two years. The conditional probability table is constructed using these observations as shown in Table 4.11.

4.7.3 Computation of Priors for Events

From two years of traffic alerts collected at various locations in Delhi, we compute the prior probability of traffic related events. For instance, there may be many reasons for delays at a node. These events occur at different probabilities at different nodes. One node may be prone to a particular type of event compared to other nodes. We observed that (a) breakdowns caused majority of the delays compared to any other events which is unlikely in developed countries, (b) each location is

susceptible to certain types of events, and (c) not all locations will have events.

4.7.4 Propagate impact

The delay impact at a node (location) will affect delay at its neighboring node since the nodes are interconnected to each other by links. For computational efficiency we limit the impact propagation from a link to the previous link only i.e., if traveling from s_1 to s_2, the delay at s_1 is influenced only by the delay at s_2. This is a realistic assumption in the traffic domain. In other words, we assume that *delay* at a node s_i depends on node s_{i+1} and not on any other nodes in the route network.

4.7.5 Illustration of Impact Assessment

We will assess the impact of the first message in Table 4.10 and compute the probability of delay.

4.7.5.1 Qualify impact

In Figure 4.17, a complete example from SMS notification to event extraction is shown, followed by automatic mapping of its locations to their closest stop names for transport network in GTFS. In the running example, the locations are: $e^i_{loc_{start}}$ = *Dhaula kuan*, $e^i_{loc_{end}}$ = *Moolchand*, $e^i_{loc_{on}}$ = *Raj Nagar flyover*.

Accident	BreakDown	delay
0	0	0
1	0	1
0	1	1
0	1	0
1	1	1
0	1	1
0	0	0

	Accident	Yes		No	
	BreakDown	Yes	No	Yes	No
delay	Yes	1	1	0.667	0
	No	0	0	0.333	1

Table 4.11: Top and bottom tables show the Traffic Observations and the Conditional Probability Table (CPT) respectively for the Bayesian Network constructed for Dhaula Kuan location which is represented as a node in our route network.

4.7.5.2 Quantify impact

The impact assessment of an event from textual observation on the schedule is done using the parameterized background knowledge. We will compute the probability of delay at Dhaula kuan given the example message.

$$P(delay \mid BreakDown = \text{“}Yes\text{”}, Accident = \text{“}No\text{”}) = \frac{P(BreakDown = \text{“}Yes\text{”}, Accident = \text{“}No\text{”} \mid delay)P(delay)}{P(BreakDown = \text{“}Yes\text{”}, Accident = \text{“}No\text{”})}$$

From Table 4.11 we consider observations for which we have a breakdown and no accident for computing all the probabilities. There are four instances for P(BreakDown = “Yes”, Accident = “No”, delay = “true”). Hence, the first probability in the numerator is (1/2). There are seven total instances of observations out of which four have delays. Hence the second probability in the numerator is (4/7). Out of seven total instances, there are three instances for which there is a breakdown but no Accident. Hence the denominator in the above equation is (3/7). $P(delay \mid BreakDown = \text{“}Yes\text{”}, Accident = \text{“}No\text{”}) = (1/2)\ (4/7)/(3/7) = 0.667$. The same process is used to build the CPT shown in Table 4.11. This shows that given a breakdown event at *Dhaula Kuan*, we are 66.7% confident that there will be a traffic delay.

We acknowledge that the illustration above did not fully utilize the power of Bayesian network. The proposed approach can be easily extended for adding multiple modalities of observations as they become available.

4.8 System Architecture

Figure 4.19 depicts the overall system architecture of the multimodal dynamic update system. We use the SMSGupShup API[15] to get all the SMS alerts on an hourly basis (setup as a cron job). For each message, we extract the event related metadata - location (from, to, on), time, and event

[15] http://api.smsgupshup.com/api

(type). Event metadata such as location name and time are used to find the affected stops using stop names and trip information. Trip information is time stamped route for a public transport vehicle and it has a direction. Event extraction is run over historical traffic alerts for two years, and for ten cities in India. The event related priors are computed for all the cities and the city (Delhi) specific priors are computed in the learn parameters step. Once the Bayesian network is parameterized, it can be used to reason over new traffic related observations for estimating the probability of having a delay at a node.

4.9 Evaluation: Traffic Event Extraction from SMS Messages

We describe the characteristics of the SMS updates from SMSGupShup and present our evaluation on the data collected for two years for the city of Delhi.

4.9.1 Dataset

Delhi Traffic Police (DTP) deploys units on roads of Delhi for monitoring traffic conditions. These units report traffic incidents as they occur (and confirmed) in the city. Thus, the number of alerts sent out on a particular day depends on the number of incidents in the city. We collected around 9,000 SMS alerts (on which we evaluate our approach) for the city of Delhi for two years which is on an average, 12 alerts a day. We found that the average alerts per day is not necessarily how it is accumulated in practice, e.g. on a rainy day in Delhi, a location may have more than 12 alerts while no alerts at all on other days.

4.9.2 Reasoning Over Traffic Events

The evaluation presented in Table 4.12 answers some of the questions posed regarding events in the quantitative reasoning section. Each location may have multiple independent events and each event

Location	$\hat{P}$(delay | BreakDown)	$\hat{P}$(delay | Weather)	$\hat{P}$(delay | Procession)	$\hat{P}$(delay | RepairWork)	$\hat{P}$(delay | Accident)
Lajpat Nagar	**0.0199**	0.016	0.0039	0.0017	0.007
Dhaula Kuan	*0.0774*	0.028	*0.0748*	*0.0160*	*0.031*
Munirka	0.006	0.008	**0.0118**	0.0035	0.0077
Ashram	**0.0553**	0.032	0.0039	0.0035	0.0233
ITO	0.0110	**0.012**	0.0039	0.0071	0.0038
Chirag Delhi	0.0243	*0.032*	0.0196	0.0089	0.0077
Sansad Marg	0.0066	0.012	**0.0275**	0.0053	0.0116
Moti Bagh	**0.0221**	0.012	0.0118	0.0142	0.0116
Noida	**0.0309**	0.018	0.0177	0.0053	0.0038
Faridabad	**0.0044**	0.004	0.0039	0.0017	0.0038

Table 4.12: Delay probability estimation for ten locations in Delhi computed using the priors from the traffic alerts from ten cities.

may have independent probabilities with which they occur at different locations. Given a location, the delay probability varies with event types, e.g., at Lajpat Nagar, BreakDown event causes greater delays compared to Accident. BreakDown affects delays at Lajpat Nagar and Dhaula Kuan with different intensities which can be explained by probabilities $\text{P}(\text{Weather}_{DhaulaKuan} \mid \text{delay}) = 0.0588$ and $\text{P}(\text{Weather}_{LajpatNagar} \mid \text{delay}) = 0.2352$. This indicates that given evidence of a delay, the cause being bad weather is 23% at Lajpat Nagar. Similarly, given delay as evidence, bad weather being the cause is 5% at Dhaula Kuan.

4.10 Discussion

We will reflect on some of our observations while utilizing SMS messages for extracting traffic events.

4.10.1 SMS as a mode of traffic alerts

The SMS alerts are a convenient and preferred way of conveying real time road traffic conditions [31]. Many cities have realized this and they provide traffic updates and road conditions through SMS alerts. With the data collected for two years and the reasoning carried out with new incoming

traffic alerts, we showed the feasibility of computing delay probabilities. The delay probability can be consumed by city administration for better planning and allocation of resources.

The event extraction from SMS alerts needs a careful consideration since it directly influences the inference process. Event extraction is used for processing both historical and current observations. Using recurring patterns for determining location and event types resulted in good results.

4.10.2 Sample space for computing probabilities

Probabilities presented in Table 4.12 were computed considering each day as one unit in the sample space. For example, if there were accidents on 4 days out of 30 days at a location, then we consider 30 days as our sample space. This is reasonable in the domain of traffic, since we can accumulate knowledge of the prior probability of accidents at this location over time. We found that some locations are prone to particular types of events more than others thereby influencing the prior probabilities of events associated with them.

4.10.3 Relative ordering of delay probabilities for decision making

The relative ordering of delay probabilities is important rather than the absolute value of the delay probability for decision makers. For instance, given that there is going to be a heavy rain in Delhi, which location is affected the most? The location named *Chirag Delhi* is the worst affected location by weather related events. Similarly, given that there is a procession in the city of Delhi, *Sansad Marg* is the worst affected location followed by *Munirka* (from Table 4.12). Procession, repair work, accidents, and break downs affect *Dhaula Kuan* the most, and out of all the locations presented here, it is the location worst hit by traffic delays. This information is valuable for the deployment of resources such as traffic personnel, city cops, traffic diversion, public transport schedule updates, etc.

4.10.4 Consumption of delay probabilities

The route recommendation system IRLTransit uses the delay probabilities in ordering the route recommended to the commuters. This ordering is based on the relative ordering of delays. For those nodes without any probability of delay estimates due to missing or absence of events at the location, we take the delay probability estimate as zero. Figure 4.15 shows the route recommendation by considering the dynamic updates on traffic events.

4.10.5 General Observations

We could estimate the lower bound on the probability of delays given an event at a location. The interaction between events, i.e., two or more events co-located in space and time is unknown from the given observations. This will be useful in estimating delay probabilities in the cities having multiple events at a single location.

We proposed a solution framework for processing dynamic traffic updates and estimation of the delay probability for each location. We addressed many challenges such as absence of machine sensors for monitoring traffic conditions, extraction of location and event observations from unstructured text, representation and parameterization of a Bayesian network using prior knowledge, dealt with sparse observations, and defined impact propagation. We evaluated our approach using real world dynamic updates on traffic events to recommend solutions leading to timely schedule information of public transport vehicles.

5

Event Understanding

"Any fool can know. The point is to understand."

— Albert Einstein

We uncover various interactions among events in the physical world and model their evolution over time in this Chapter. Structure of a graphical model provides insights into complex interactions among various events in the physical world. Further, to deal with dynamism, we utilize graphical models that captures evolution of traffic dynamics over time. We present techniques to synthesize normal traffic dynamics from massive time series data. Based on the normalcy models, we tag anomalies in traffic dynamics and for better understanding, we interpret/explain anomalies using traffic related events from textual data.

Graphical models have been successfully used to deal with uncertainty, incompleteness, and dynamism within many domains. These models when built bottom-up from data ignores pre-existing declarative knowledge about the domain in the form of ontologies and Linked Open Data (LOD) that is increasingly available on the web. In this Chapter, we present an approach to leverage such "top-down" domain knowledge to enhance "bottom-up" building of graphical models. Specifically, we propose three operations on the graphical model structure to enrich and/or rectify the structure with additional nodes, additional edges, and modified edge directions. We illustrate the enrichment

process using traffic data from 511.org and declarative knowledge from ConceptNet. The resulting enriched graphical model can potentially lead to higher fidelity and better predictions of traffic delays.

5.1 Understanding Events Utilizing Social and Sensor Data

Understanding speed and travel-time dynamics in response to various city related events is an important and challenging problem. Sensor data (numerical) containing average speed of vehicles passing through a road link can be interpreted in terms of traffic related incident reports from city authorities and social media data (textual), providing a complementary understanding of traffic dynamics. State-of-the-art research is focused on either analyzing sensor observations or citizen observations; we seek to exploit both in a synergistic manner.

We demonstrate the role of domain knowledge in capturing the non-linearity of speed and travel-time dynamics by segmenting speed and travel-time observations into simpler components amenable to description using linear models such as Linear Dynamical System (LDS). Specifically, we propose Restricted Switching Linear Dynamical System (RSLDS) to model normal speed and travel time dynamics to approximate nonlinearity by piecewise linear approximation over different time intervals and thereby characterize anomalous dynamics. We utilize the city traffic events extracted from text to explain anomalous dynamics. We present a large scale evaluation of the proposed approach on a real-world traffic and twitter dataset collected over a year with promising results.

5.1.1 Preliminaries

We define representation of city traffic related events and the road network. We also provide a brief overview of Linear Dynamical System (LDS) and propose Restricted Switching Linear Dynamical System (RSLDS) to characterize traffic dynamics.

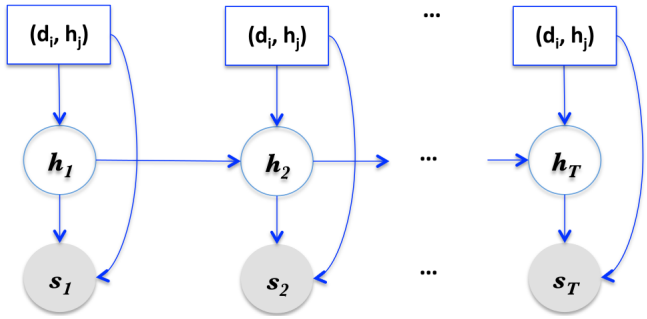


Figure 5.1: A Restricted Switching Linear Dynamical System (RSLDS) with each switch variable indexed by day of week and hour of day (d_i, h_j).

5.1.2 Traffic Event Representation

We use a quintuple $\langle \hat{e}_t, \hat{e}_l, \hat{e}_{st}, \hat{e}_{et}, \hat{e}_i \rangle$ to represent an event where, $\hat{e}_t$ represents the event type, $\hat{e}_l$ is the location of the event, $\hat{e}_{st}$ is the start time of the event, $\hat{e}_{et}$ is the end time of the event, and $\hat{e}_i$ represents the estimated impact of the event. We use a unified representation of events for both 511.org reported events and traffic events extracted from twitter.

5.1.3 Road Network

The fundamental building block of a road network is called a *link*, represented by l. 511.org provides location information for all the links in San Francisco Bay Area. A road r, is an ordered sequence of links, i.e., $r = [l_1, l_2, ..., l_n]$, where, n is the number of links in road r. The location of a link is specified by start and end lat-long which can be used to reconstruct the road. We collect speed and travel time observations from 511.org for 3,622 links. The whole road network N is a set of roads, $N = \{r_1, r_2, ..., r_m\}$, where m is the total number of roads in the road network.

5.1.4 Problem Formulation

Speed and travel time dynamics in the domain of traffic follows a more or less weekly recurring pattern based on the hour of the day and the day of the week. Traffic dynamics may vary abnormally due to various city traffic related events, varying road conditions, and random effects. We are not

guaranteed to have access to all the active city traffic related events and their interactions. A Gaussian Mixture Model (GMM) approach to model speed and travel time variations [135] do not capture the temporal dependencies fundamental to traffic dynamics. Time series techniques such as autoregressive (AR) and autoregressive-integrated-moving-average (ARIMA) models [86; 104] capture temporal dependencies. However, relating later values of speed and travel time with corresponding earlier values alone is not adequate as they cannot capture the latent factors effecting traffic and hence crucial in modeling traffic dynamics. An LDS model offers a better foundation for representing these additional factors that are difficult to capture separately in modeling traffic dynamics. The volume of vehicles through a link, associated interactions, and random effects (noise) on traffic dynamics are being lumped together and approximated by the hidden (latent) states ($h_{1:T}$) of the LDS model and the noise terms $\eta_{\mathbf{t}}^{\mathbf{h}}$ and $\eta_{\mathbf{t}}^{\mathbf{s}}$ as shown in Figure 3.5 (Note that we do not have adequate knowledge to separately account for the influences of various latent factors and hence the recourse to lumping). The average speed of vehicles passing through a link and the average travel time for a link, obtained from sensor data are represented by the observed node ($s_{1:T}$) in Figure 3.5.

We broadly categorize various factors that influence traffic into *internal* and *external* factors. *Internal* factors include day of the week, time of the day, and location. *External* factors include city traffic related events such as accidents, breakdowns, music and sporting events. We propose a *Restricted Switching Linear Dynamical System* (RSLDS) as shown in Figure 5.1. We learn one LDS model for each hour of the day and for each day of the week, giving us 24×7 (168) LDS models for each *link*. A switch variable in RSLDS is used to index and select an LDS model on (d_i, h_j), where, d_i is day of week (ranging over 7 days) and h_j is hour of day (ranging over 24 hours). Our approach is similar to Switching Linear Dynamical System (SLDS) [120] that allows discrete switches to select an appropriate LDS model. However, SLDS model assumes a Markovian transition between switch configurations, which is violated in the domain of traffic. For example, the *external* factors such as accidents and breakdowns may occur randomly and independently.

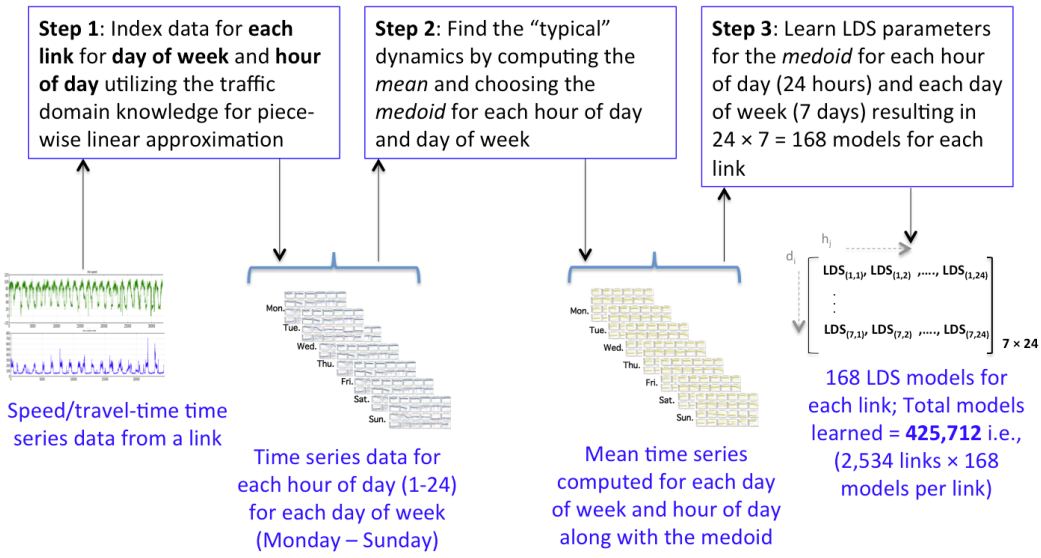


Figure 5.2: Learning normal traffic dynamics from speed and travel time observations resulting in 168 LDS models for each link in the road network.

5.1.5 Approach

We present our approach to learn models for normal traffic dynamics, tagging anomalies, and utilizing events from textual stream to explain the anomalies below.

5.1.6 Learning Normalcy in Traffic Dynamics

Figure 5.2 outlines the process of learning normal traffic dynamics. LDS is a linear model and cannot faithfully capture the non-linearity in speed dynamics over time. RSLDS deals with non-linearity by piecewise linear approximation using a collection of LDS models by selecting appropriate linear regime from the collection based on the switch state. Hour of the day has a major influence on traffic dynamics, e.g., morning peak hours (7 am to 9 am) and evening peak hours (5 pm to 7 pm) on a work day typically has slow moving traffic[1]. Day of the week is another important influencer of

[1]Contrary to this knowledge, we observed a single dip only during evening peak hours for a link. Upon further investigation, we found that it was a one-way street leading out of downtown. Effectively, we were able to uncover a policy (one-way) from observational data.

traffic dynamics, e.g., weekend pattern is different as offices are closed but social, music, or sporting events may occur at certain locations and time durations. We use both the day of the week and the hour of the day to index traffic dynamics and learning normalcy model.

5.1.6.1 Indexing Traffic Dynamics Data

In Step 1 of Figure 5.2, we partition data from a link based on the day of week (Mon-Sun) and further, on the hour of the day (1-24). Hourly speed dynamics for each Monday between May 2014 and Jan 2015, and each hour is shown in Figure 5.4. Each of the 24 subplots corresponds to the time series of speed variation over each hour of the day. We observe approximate clustering of speed dynamics (light colored lines) in most of the plots, indicating a general hourly trend in speed dynamics. In the first seven hours of the day, starting 12 AM to 7 AM, the average speed of the vehicles remain high and stable, around 80 to 100 km/h. After 7 AM, we observe a decreasing trend in speed until 9 AM, which may be due to morning commute. After an increasing trend in speed around 10 AM, possibly due to subsiding commuter traffic, the speed of vehicles is observed to be stable from 11 AM to 1 PM. A decreasing trend is observed between 1 PM and 2 PM with speeds plummeting to 20 to 30 km/h after 2 PM until 6 PM. This can be due to lunch time and evening rush hour traffic respectively. Closer to 7 PM, peak hour rush subsides resulting in increasing speed trend till 8 PM. After 9 PM, the speed resumes and stabilizes between 80 to 100 km/h. While driver behavior and road speed-limit dictate speeds at normal times, the high occupancy of the road due to traffic related events causes delays at other times.

5.1.6.2 Selecting Typical Traffic Dynamics

In Step 2 of Figure 5.2, we select typical traffic dynamics by iterating through the index of Step 1. Algorithm 3 describes the selection of a typical traffic dynamics for each hour. The input to Algorithm 3 is the speed observations indexed over *internal factors*. Each hour contains multiple speed sequences $[s_{(m,1)}, ..., s_{(m,n)}]$ (if there are five Mondays with 60 observations for an hour, then,

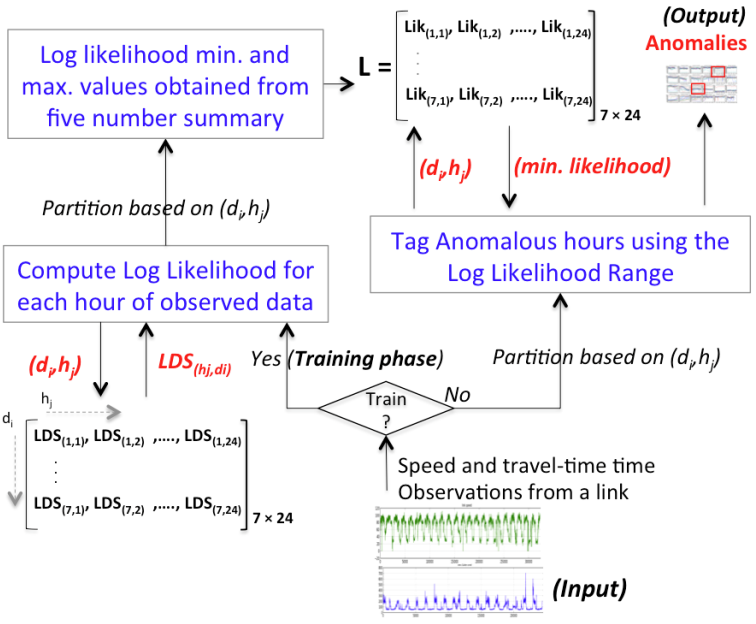


Figure 5.3: Utilizing 168 LDS models to tag anomalies, which can be tied to a city event reported on textual stream.

$m = 1$ to 5 and $n = 60$; speed observations are sampled n times an hour to create each of the m sequences). For computing the average speed at each of the n sampling point, we sum up all the speed values at each sampling index (1 to n) over all the m sequences and divide it by m. Average speed sequence serves as the centroid of all the speed sequences. To select a speed sequence that exists in the real-world (that is, it is *realizable*), we choose the speed sequence that is closest to the centroid using a point-wise Euclidean distance metric, obtaining the medoid.

The result of running Algorithm 3 is shown in Figure 5.4 with mean speed plot (dashed line) and medoid (solid line).

5.1.6.3 Learning LDS parameters

In Step 3 of Figure 5.2, we learn the parameters of the LDS utilizing the representative traffic dynamics chosen based on Algorithm 3. The LDS parameters are learned for every day of week and every hour of day. LDS is parameterized by $\theta = \{\mathbf{A}, \mathbf{\Sigma_h}, \mathbf{B}, \mathbf{\Sigma_s}, \mu_\pi, \mathbf{\Sigma}_\pi\}$ where $\mathbf{A}$ is the transition

Algorithm 3: Select medoid for hourly speed plots

Require: Multiple speed observation sequences collected for each (d_i, h_j) where d_i = Monday to Sunday and $h_j = 1$ to 24, each set containing n speed observations, $[s_{m,1}, ..., s_{m,n}]$ where, m indexes over number of speed sequences collected for (d_i, h_j)

Ensure: $[s_1 ..., s_n]$ representing the medoid for each $M(d_i, h_j)$

for each day d from Monday to Sunday **do**

for each hour h of the day ranging from 1 to 24 **do**

Select speed values $[s_{(m,1)}, ..., s_{(m,n)}]$ from (d_i, h_j)

Find the average speed $[a_1 ..., a_n]$ from m samples

Select speed sequence $[s_1 ..., s_n]$ closest to average

Set $M(d_i, h_j) = [s_1 ..., s_n]$

end for

end for

matrix, $\mathbf{\Sigma_h}$ is the transition covariance, $\mathbf{B}$ is the emission matrix, $\mathbf{\Sigma_s}$ is the emission covariance, μ_π and $\mathbf{\Sigma}_\pi$ are the mean and covariance of the initial state density $p(h_1)$ in Equation 3.11. Since the joint distribution of LDS contains hidden variables, Expectation Maximization (EM) algorithm is used for learning the LDS parameters [54; 15]. From Equation 3.11, the joint distribution can be rewritten as

$$\begin{aligned} \ln p(h_{1:T}, s_{1:T}|\theta) = \ln p(h_1|\mu_\pi, \mathbf{\Sigma}_\pi) \quad + \\ \sum_{t=2}^{T} \ln p(h_t|h_{t-1}, \mathbf{A}, \mathbf{\Sigma_h}) + \sum_{t=1}^{T} \ln p(s_t|h_t, \mathbf{B}, \mathbf{\Sigma_s}) \end{aligned} \tag{5.1}$$

Equation 5.1 has explicit parameterization and represents the log likelihood of data given the parameters. EM algorithm chooses the initial parameters θ^{old} and evaluates $p(h_{1:T}|s_{1:T}, \theta^{old})$ in the expectation step. In the maximization step, the expectation of the log likelihood function represented by $\mathbb{E}_{h_{1:T}|\theta^{old}}[\ln p(h_{1:T}, s_{1:T}|\theta)]$ is maximized with respect to θ. The parameters are updated to $\mathbf{A^{new}}, \mathbf{\Sigma_h^{new}}, \mathbf{B^{new}}$, $\mathbf{\Sigma_s^{new}}, \mu_\pi^{\mathbf{new}}$, and $\mathbf{\Sigma}_\pi^{\mathbf{new}}$. After maximization step, if the convergence criteria is

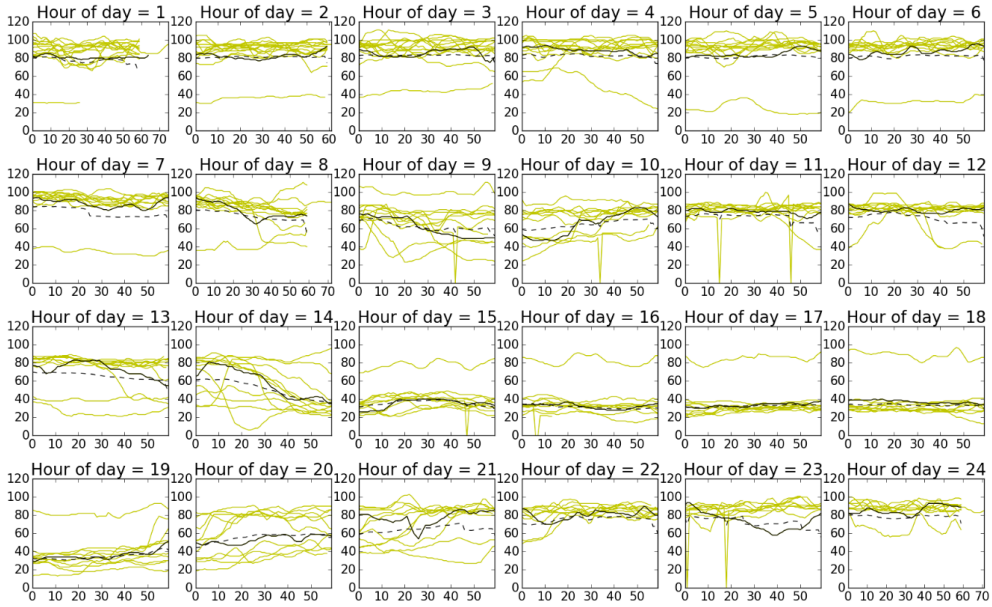


Figure 5.4: Hourly plot of speed variations over time for all the Mondays from May 2015-June 2015 for a link.

not satisfied, the new parameter setting for LDS is computed, θ^{new} replaces θ^{old} and the algorithm returns to the expectation step.

5.1.7 Detecting Anomalous Traffic Dynamics

Figure 5.3 outlines the process of learning normalcy in terms of log likelihood score and utilizing it to tag anomalies. In the training phase, we utilize the 168 LDS models learned for each link indexed by *internal factors*, $\theta(d_i, h_j)$, for estimating the log likelihoods, $\ln p(h_{1:T}, s_{1:T}|\theta(d_i, h_j))$ as shown in Equation (5.1). In the training phase, we learn the typical likelihood values after aggregating the log likelihood scores for the entire dataset partitioned by (d_i, h_j), thereby capturing normalcy. We utilize a non-parametric approach of five number summary (*minimum*, first quartile, median, third quartile, and *maximum*) over the log likelihood scores for each partition indexed by (d_i, h_j). The log likelihood range (*minimum* and *maximum*) exists for each day of the week and the hour of the day, (d_i, h_j). In the testing phase, we compute the log likelihood score for the observed data

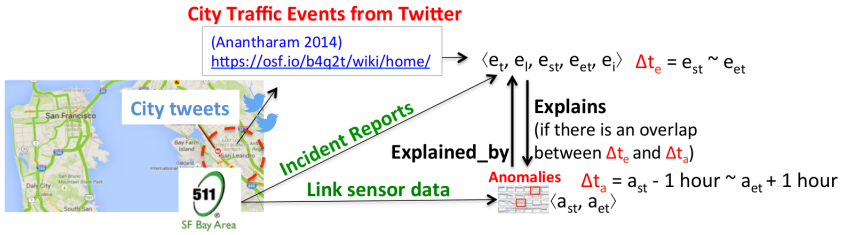


Figure 5.5: City traffic events (textual data) used to explain anomalies in traffic dynamics (sensor data).

using the appropriate LDS model $\theta(d_i, h_j)$. If the log likelihood value for a particular day of week and the hour of the day is less than the *minimum* log likelihood value (retrieved from matrix $\mathbf{L}$ in Figure 5.3), we tag the traffic dynamics as anomalous.

5.1.8 Traffic Events for Explaining Anomalies

For every city traffic event collected from textual data, we detect anomalies in traffic pattern as outlined in Figure 5.5. We examine the city traffic events using their location, start time, and end time. Based on the location of the event, we select links within a radius r km from the event. We run the anomaly detection step on the temporally selected data points from these links. If an event from the textual stream has a corresponding anomaly in the link data, we hypothesize that the event *explains* the anomaly and the anomaly is *explained by* the event. Algorithm 4 determines textual events E, that explains anomalies in sensor data. The radius r is an input parameter which can be changed but it is set to 0.5 km in our experiments. The adjusted duration of an event, $\Delta t_e = (\hat{e}_{st} - h, \hat{e}_{et} + h)$, where, h is set to 1 hour (lowest granularity of our analysis), is used to select sensor data from all the links within the radius of r km from the event location, $\hat{e}_l$. If the selected link data has anomalies, possibly explained by the textual event, then the event is accumulated in $E_{explained}$ as shown in Algorithm 4.

Algorithm 4: Explaining Traffic Events by Anomalies

Require: Set of city traffic events E containing event tuples $\langle \hat{e}_t, \hat{e}_l, \hat{e}_{st}, \hat{e}_{et}, \hat{e}_i \rangle$, latitude and longitude of all the 3,622 links in the road network, log likelihood range matrix $\mathbf{L}$ indexed by (d_i, h_j), radius parameter r km to select the links, and time parameter h for adjusting event duration

Ensure: $E_{explained}$ containing all events with corresponding anomalies in sensor data

```
for each event quintuple ⟨ê_t, ê_l, ê_st, ê_et, ê_i⟩ in E do
    Find hourly time range Δt_e = (ê_st − h, ê_et + h)
    Let M be all the links within the radius of r km from the event location ê_l
    for each link l ∈ M do
        Select data for link l filtered by duration Δt_e
        H(d_i, h_j) ← Compute the hourly log likelihood on selected data using Equation (5.1)
        if hourly log likelihood H(d_i, h_j) < minimum log likelihood from L(d_i, h_j) then
            E_explained ← ⟨ê_t, ê_l, ê_st, ê_et, ê_i⟩
        end if
    end for
end for
```

5.1.9 Evaluation

We conducted a large scale evaluation of our approach on real-world traffic sensor and twitter data collected for a year.

Traffic Dataset from 511.org and Tweets: We collected 1,638 city traffic related events from 511.org and we extracted 39,208 city traffic events from over 20 million tweets collected from May 2014 to May 2015 for San Francisco Bay Area, utilizing an openly available city traffic event extraction tool [118], resulting in a total of 40,846 city traffic events. 511.org also provides minute by minute speed and link travel time data for 3,622 links resulting in over 1.4 billion time series data

Source	Total Events	No Links	Missing Data	No Anomalies	Anomalies
511.org	1,638	201	901	145	391
Twitter	39,208	36,436	1942	18	812

Table 5.1: Evaluation results for all the events using Algorithm 4 with parameter setting: $h = 1$ hour and $r = 0.5$ km.

points. Out of 3,622 links, 1,088 links do not have any data points for the entire year. Further, there are partially missing data points in the time series for the remaining 2,534 links.

Evaluation Strategy: Traffic events from 511.org are reliable since it is reported by city authorities. We use these events as a reference in our evaluation. Algorithm 4 iterates over 1,638 events from 511.org to explain anomalies in traffic. We evaluate Algorithm 4 for finding 511.org event manifestations in sensor data. Further, we extend the evaluation to 39,208 events extracted from twitter and report our results.

Evaluation over 511.org Traffic events: We evaluate our approach by analyzing the co-occurrence of the event in textual data with the anomaly detected in the sensor data. Table 5.1 presents the evaluation summary for all the 1,638 511.org events and 39,208 twitter traffic events. Events with no links near them (for $r = 0.5$ km) are placed under *No Links*. If there are links near an event, with data that may be missing (for the event duration), then they are characterized as *Missing Data*. Links near an event with data are used to tag anomalies and the result is placed under *No Anomalies* and *Anomalies*. We call the events corroborated with anomalies in any of the link sensor data near the event as being *explained*. Events without accompanying anomalies in any of the link sensor data are called *un-explained*. For a palatable comparison, we present percentages of events from 511.org and twitter that explain anomalies in Figure 5.6. We observe a larger set of links near events from 511.org relative to twitter events as shown in Figure 5.6 (bottom). Out of 33% 511.org events with complete sensor data, we could explain 72% of them. Figure 5.6 (top) presents a sample output of Algorithm 4 after processing 10 events. For events marked in bold, we found anomalies in

traffic dynamics possibly explained by the event with the following insights: (a) Long-term events may not manifest as anomalies in sensor data. RSLDS normalcy model is trained over the entire year of average speed and travel time observations. Long-term events such as construction activities may span several months. Since the data we have is for a year, such long-term events are part of the normalcy model and may not be tagged anomalous as observed in Figure 5.6 (top). Events, such as accidents and disabled vehicles, are short lived events that may manifest as anomalous traffic. (b) Location and start time of the event may impact its manifestation in sensor data. Events near crowded places would most likely manifest as anomalies. Events occurring during off-peak hours are less likely to manifest in sensor data compared to the events occurring during peak-hours. (c) Missing data creates challenges for associating anomalies with events. Among the 2,534 links with data, there is missing data for many days in a year due to maintenance and sensor failures resulting in decreased coverage.

Evaluation over Twitter Traffic Events: Twitter traffic events are dispersed widely across the city resulting in reduced or missing links near many events. There are 36,436 twitter traffic events with no links near them as shown in Table 5.1 due to significantly lower sensor data coverage. Consequently, we observe that the coverage can be significantly improved by augmenting information from sensor data with that from twitter events as shown in Figure 5.6 (bottom). We expected a higher twitter traffic event manifestation in sensor data since people will most likely report events of significant impact while 511.org reports all possible traffic related events that may have varying impact. Out of 2% twitter traffic events with complete sensor data, we could corroborate 97% of it with anomalies.

Scalability Challenges: There are 2,534 links with data. For each link, we learn 168 LDS models by analyzing over 1.4 billion data points resulting in a total of 425,712 (= 2,534 $\times$ 168) LDS models. The size of the traffic dataset is around 30 GB. Learning LDS parameters and the criteria for anomaly is computationally expensive. For each link with one year of data, we estimated 25 minutes for learning LDS models and 15 minutes for computing the criteria for anomaly, resulting in a total

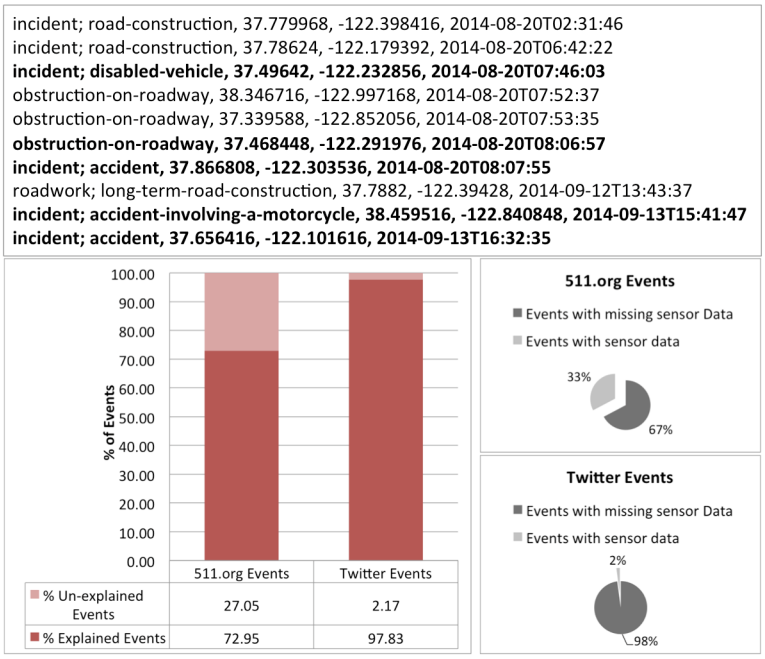


Figure 5.6: City traffic events with available and missing link data; 511.org events has higher link data availability of 33% compared to 2% link data availability for events extracted from twitter. For those traffic events with complete link data available, the bar graph shows the percentage of events explained by an anomaly as explained in Algorithm 4.

processing time of 40 minutes per link. Extrapolating processing time for all the links, we get 1,689 hours ($\frac{40minutes \times 2{,}534}{60minutes}$) (≈2 months). Initial processing was done with 2.66 GHz, Intel Core 2 Duo processor with 8 GB main memory. We then exploited inherent "embarrassing parallelism" to devise a scalable implementation of our approach on Apache Spark [154] that takes less than a day. The Apache Spark cluster used in our evaluation has 864 cores and 17TB main memory.

Normal traffic dynamics can be captured using RSLDS, a variant of LDS model, that utilizes domain knowledge to segment nonlinear traffic dynamics into linear components. Utilizing the normalcy model, we could explain anomalies in traffic sensor data using traffic events from textual data. We could also associate real-world events that impact traffic by determining anomalies in traffic pattern. Further, a large scale evaluation of our approach on a real-world dataset collected

for a year (where sensor data was available) corroborated 72% of 511.org events and 97% of twitter traffic events in terms of anomalous traffic dynamics. In future, RSLDS model capturing temporal dynamics can be utilized to study various traffic event types and associated speed and travel time dynamics (signature of an event) and predict traffic dynamics based on the traffic events from textual streams.

5.2 Understanding Associations between Events

Cyber-Physical Systems (CPS), which have tightly coupled computation, communication, and control components, are increasingly being used in various domains [128]. The observations and interactions in a CPS are characterized by: (1) *incompleteness* due to partial observation from the real world, (2) *uncertainty* due to inherent randomness involved in the sensing process (noise in case of machine sensors and bias in case of citizen sensors), and (3) *dynamism* from the ever changing and non-deterministic conditions of the physical world. Graphical models can be used to deal with incompleteness, uncertainty, and dynamism in many diverse domains such as traffic management, healthcare, system health monitoring, speech processing, image processing, and computational biology [111; 133; 4; 134; 22]. These models are built bottom-up using sensor data in most cases, i.e., the structure (conditional independence between variables) and parameters (Conditional Probability Distribution – CPD for continuous variables, or Conditional Probability Tables – CPT for discrete variables) are learned from sensor data [77]. Some of these models also allow domain experts to express their declarative knowledge of the domain in the form of value constraints and inequality constraints in the learning process [94].

Extracting structure of a graphical model from observations of a CPS is very challenging due to data sparsity, incompleteness, and difficulty in detecting causal links. However, declarative domain knowledge can obviate the need to learn everything from data. In addition, correlations derivable from data can be further consolidated if the declarative knowledge base provides evidence that it

is causal in nature. For example, the influence of a music event on traffic flow may be known a priori. However, this declarative knowledge is insufficient to answer quantitative questions which are addressed by graphical model parameters.

Declarative knowledge (including causal relationships) is increasingly being published using open data standards on the Semantic Web [25]. These include knowledge bases such as ConceptNet5 [90] and many domain ontologies and data sets published on the Linked Open Data (LOD) [24] cloud. We hypothesize that leveraging such knowledge will increase the fidelity of graphical models. More importantly, it will complement structure learning algorithms of graphical models by utilizing declarative domain knowledge. Specifically, we focus on how knowledge gaps (incompleteness) can be reduced.

The contributions of this work include:

- definition of external events and internal observations that co-exist across CPS,
- extraction of graphical model structure by correlating external events (e.g., music concert) to internal observations (e.g. delays),
- definition of three operators (addition of nodes, addition of edges, and modification of edge directions) for complementing structure learning of graphical models using declarative domain knowledge (e.g., ConceptNet), and
- evaluation of the approach on a real world traffic data set (511.org).

The enriched model is more encompassing of the domain variables and relationships between them, which can potentially lead to better prediction of delays.

5.2.1 Preliminaries

We define notations and terminologies for representing sensor observations and define abstractions over these observations which are used in rest of the Chapter. Recall that we categorize the traffic

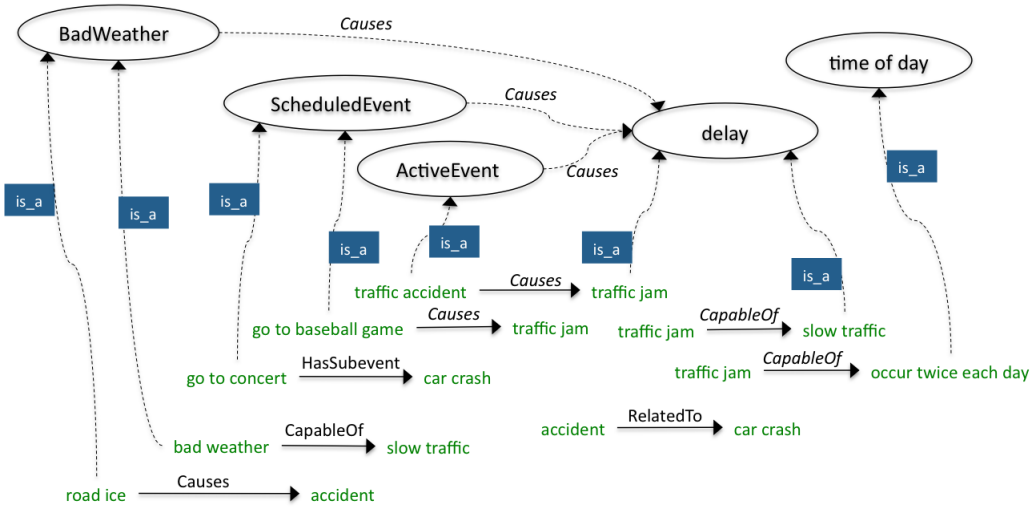


Figure 5.7: Domain knowledge of traffic in the form of concepts and relationships (mostly causal) from ConceptNet (causes is to be interpreted as can cause)

related observations into two major categories: (1) Internal observations, and (2) External events. Internal observations consists of observations from on-road sensors monitoring flow of traffic. They are called “internal" since these observations are internal to the road network, e.g., speed, volume, and travel time. External events constitutes all the events external to the road network that may or may not influence traffic. Scheduled and active events constitute external events.

5.2.2 Notations

A link is a fundamental building block of a road network. Multiple links connected back to back constitute a road. Each link is monitored by sensors measuring speed of vehicles, volume of vehicles, and time to travel the link. Let $\langle O_s, O_v, O_t \rangle$ be the corresponding variables representing internal observations of a link. $\langle E_a, E_s, D_{week}, T_{day}, O_{delay} \rangle$ are binary variables representing active and scheduled events, day of the week, time of the day, and delay in link travel time, which are all external events. Each of these are represented as a random variable, whose value is unknown till we observe it. External events have start time (t_s) and the duration it persists (t_d, provided by

511.org), using which we compute the event end time (t_e). We generate a training set by processing each observation record (reported almost every two seconds) where O_s, O_v, and $O_t \in \mathbb{R}$ are sensor observations. E_a, E_s, D_{week}, T_{day}, and $O_{delay} \in \{0,1\}$, and are set to 1 depending on whether the external event is active. We compute the average travel time $T_{avg,l,h}$ for each link (l) for every hour of the day (h ranges for 24 hours) for total number of days (n days) considered for analysis. We represent the dataset created using tuples of the form $\langle O_s, O_v, O_t, E_a, E_s, D_{week}, T_{day}, O_{delay} \rangle_{timestamp}$

Domain knowledge from ConceptNet, shown in Figure 5.7, is represented by a set of concepts and relationships between them ($G_{declarative}$).

5.2.3 Problem

Build a graphical model G inherent in the domain of traffic consisting of random variables $\langle O_s, O_v, O_t, E_a, E_s, D_{week}, T_{day}, O_{delay} \rangle_{timestamp}$. The structure consists of connections between random variables that specifies conditional independence statements. The parameters are the CPT or CPD learned from data. Our focus is on complementing structure extraction with declarative knowledge. Parameter learning (CPT or CPD with probability values/distributions) is out of scope of this work. We address two problems: 1. Extract the structure G from traffic data. 2. Use $G_{declarative}$ to modify G. This step may add to G (a) new random variables, (b) new links, and (c) modify link directionality. This analysis provides the kind of events that can influence traffic. That is, the structure that we learn can be used to determine the events that cause delays and finding them will allow us to understand the latent factors modeled in Section 5.1.

5.2.4 Approach

We now present the details of our approach of complementing graphical model structure extraction with declarative knowledge from knowledge sources. We focus on structure extraction and refinement, using declarative knowledge and postpone parameter learning as a future work. Starting

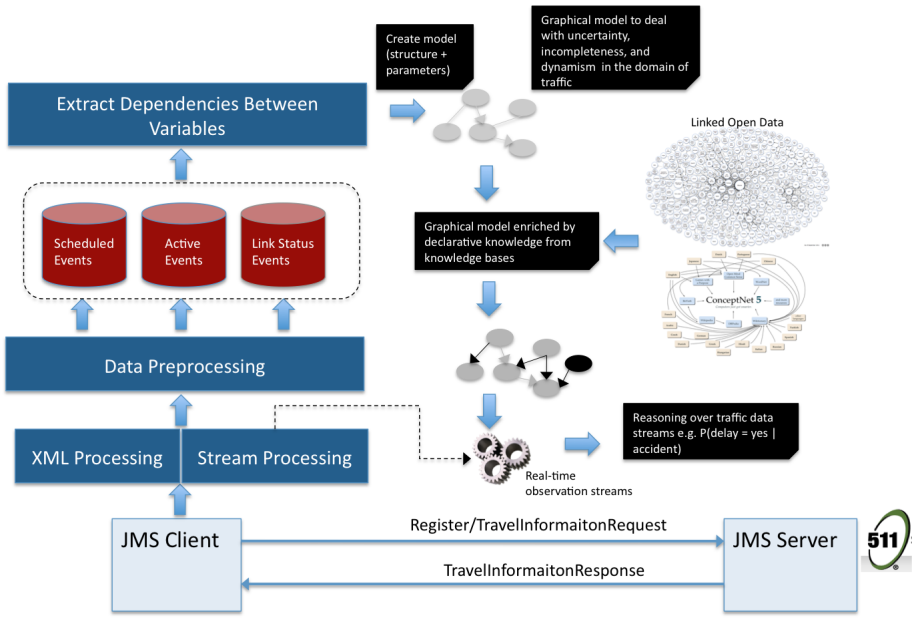


Figure 5.8: Traffic observation stream processing pipeline along with our approach to complement graphical models with knowledge from existing knowledge bases on the web

from the overall system architecture, this section provides details of the approach toward utilizing declarative knowledge in the construction of graphical models for CPS.

5.2.5 System Architecture

The overall system architecture is shown in Figure 5.8. The raw observations from traffic data feed of 511.org are preprocessed to generate training data. The training data consists of random variables used to describe the domain of traffic which includes O_s, O_v, O_t, E_a, E_s, D_{week}, T_{day}, and O_{delay}. The Java Messaging Service (JMS) is used as a mode of subscription to 511. The XML messages are parsed to extract scheduled and active events, and link status (speed, volume, and travel time) information. These observations are further processed to generate boolean abstractions such as time of day, peak hour, and day of week. This data set is used as a training input to the structure learning algorithm which extracts dependencies between the random variables.

Algorithm 5: Algorithm for preprocessing traffic data stream

1: **for** each $\langle O_s, O_v, O_t \rangle_{timestamp} \in$ TrafficStream **do**
2: $E_a = 0$, $E_s = 0$, $D_{week} = 0$, $T_{day} = 0$, $O_{delay} = 0$
3: **if** timestamp $\in$ timestamp of E_a **then**
4: $E_a = 1$
5: **else if** timestamp $\in$ timestamp of E_s **then**
6: $E_s = 1$
7: **else if** timestamp $\in$ saturday, sunday **then**
8: $D_{week} = 1$
9: **else if** timestamp $\in$ peak-hour; 7 AM to 9 AM $\vee$ 4 PM to 6 PM **then**
10: $T_{day} = 1$
11: **else if** $O_t > \mathrm{T}_{avg,l,h}$ **then**
12: $O_{delay} = 1$
13: **end if**
14: $\langle\ O_s, O_v, O_t,\ E_a,\ E_s,\ D_{week},\ T_{day},\ O_{delay}\ \rangle_{timestamp}$
15: **end for**

5.2.6 Preprocessing of Traffic Observations

Traffic observations are processed using the Algorithm 5 to generate the training data, which is then used by the structure learning algorithms to extract inherent structure in the domain of traffic. There are additional fields added in the preprocessing step derived from raw sensor observations and active and scheduled event status. The raw observations are mapped to binary values to form the training data. The timestamp of the observation is used to check for presence of active events, scheduled events, day of week, time of day, and current link delay status. The corresponding boolean variable is set to 1 depending on the presence of the event at the observation timestamp.

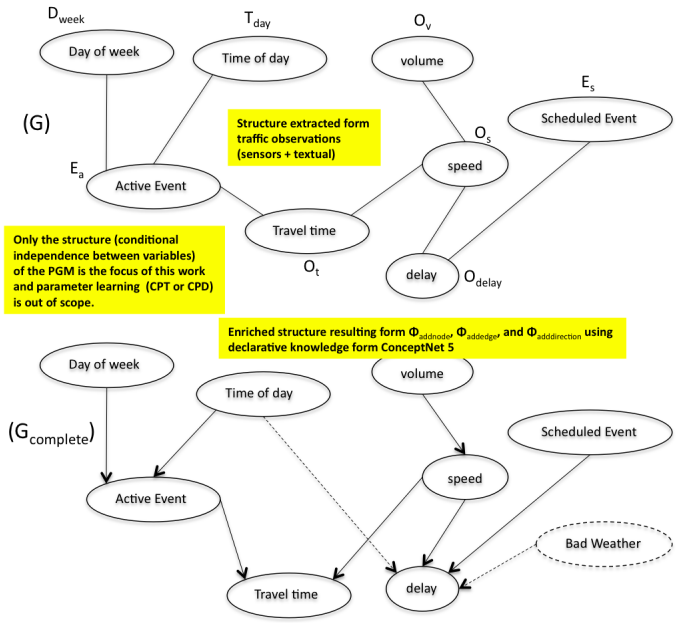


Figure 5.9: Top part of the figure depicts the structure extracted from traffic observations and the bottom part has the enriched structure (using declarative knowledge)

5.2.7 Structure Learning for Graphical Model

Figure 5.9 shows the structure extracted using observations $\langle O_s, O_v, O_t, E_a, E_s, D_{week}, T_{day}, O_{delay} \rangle_{timestamp}$. There are undirected links which need to be converted to directed links before the parameters are learned. We explored many structure extraction approaches including constraint-based, score-based, and hybrid algorithms. Constraint-based learning algorithms verify the conditional independence between variables with ideas derived from [115]. Score-based learning algorithms evaluate the structure and compute the goodness-of-fit score. Hybrid algorithms combine the aspects of constraint-based and score-based algorithms for structure extraction. These algorithms are implemented as part of the *bnlearn* [127] package of R [64], which is a statistical data analysis tool. Within constraint based, we used Grow-Shrink [95], Incremental Association, Fast Incremental Association [142], and Interleaved Incremental Association [151]. In score-based we explored Hill-Climbing and Tabu Search. Max-Min Hill-Climbing and Restricted Maximization [143] were

used to extract structure as part of hybrid learning approach. There are other constraint-based approaches such as Chow-Liu [35] and ARACNE [96] that learn the correlation graph (without any direction) between random variables. Chow-Liu approach gave us the best structure (i.e. intuitively most satisfactory) that described the traffic domain and we chose this for further refinement using declarative knowledge.

5.2.8 Knowledge Base to Complement Structure Learning

The structure learned from Chow-Liu's approach resulted in a correlation graph shown in upper part of Figure 5.9. While correlations are entirely data-driven and may or may not support semantic interpretation, the causal links which are more valuable, are much harder to extract. We hypothesize that, we can use declarative knowledge from knowledge base such as ConceptNet to infer causal links in the structure extracted by structure learning algorithms. The graph G in general is composed of nodes (N), edges (E), and direction (D) of edges i.e., G = $\langle$ N, E, D $\rangle$. We distinguish the extracted graph from the declarative knowledge graph $\mathrm{G}_{declarative} = \langle N_{dk}, E_{dk}, D_{dk} \rangle$

We focus on the causal and subsumption knowledge related to traffic from ConceptNet. We propose three operations on the graph structure G (result of structure extraction that encapsulates the sorts of updates that can be performed to graphical models as a result of using declarative knowledge) using the knowledge base, $\mathrm{G}_{declarative}$: (1) Add missing random variables $G_n \leftarrow \phi_{addnode}(G, G_{declarative})$, (2) Add missing links $G_e \leftarrow \phi_{addedge}(G_n, G_{declarative})$, and (3) Set directionality of the links $G_{complete} \leftarrow \phi_{adddirection}(G_e, G_{declarative})$. We assume that the subset of relevant declarative knowledge is given to us. The first step is to unify the variables represented in G and $\mathrm{G}_{declarative}$ using subsumption and then perform the three operations.

5.2.8.1 Node Addition

$\phi_{addnode}$: Due to lack of instrumentation, there may be missing or incomplete observations from a domain. Structure learning algorithms cannot account for such missing observations. We leverage

domain knowledge to add missing random variables to G. From $G_{declarative}$, we add all the nodes missing in G, i.e., $N_{new} = N \cup N_{dk}$.

5.2.8.2 Edge Addition

$\phi_{addedge}$: Due to data sparsity, the structure extracted may have missing links between different random variables. Purely data driven approaches may have this limitation. We use domain knowledge to add missing links to G. From $G_{declarative}$ we add all the edges missing in G, i.e., $E_{new} = E \cup E_{dk}$.

5.2.8.3 Edge Directionality

$\phi_{adddirection}$: The structure learning algorithms use information theoretic and statistical techniques to extract correlations between the random variables. While correlations give us links between the random variables, they do not provide the directionality of the links. A domain expert looking at the links can decide the directionality of the links to form causal links. We exploit such a declarative domain knowledge from knowledge bases to form causal links. For all the links in G without a direction (missing entry in D), we look for corresponding directionality in D_{dk} and add it to D. $G_{complete}$ is a graph that is a directed acyclic graph which is obtained by leveraging domain knowledge.

Cyber physical systems (e.g., road traffic network) are characterized by observations spanning textual (e.g., incident report) and numerical observations (e.g., speed of vehicles). The internal observations in the domain of traffic such as speed, volume, and travel time are affected by external traffic related events. We proposed a novel approach for leveraging domain knowledge (e.g., by reusing causal knowledge from ConceptNet) in extraction of dependencies between variables in the domain of traffic. Three operators (that add missing nodes, add missing edges, and assign edge directions) for enrichment of graphical models using declarative knowledge were defined. We exemplified the enrichment process using real world traffic data from 511.org and concepts from

ConceptNet. Our evaluation showed that combining graphical models with qualitative information present in declarative knowledge provides much richer domain model for reasoning. Specifically, it allows us to combine two complementary sources of information: (i) quantitative and correlation-based knowledge automatically synthesized bottom-up from traffic data, and (ii) manually curated qualitative and causal knowledge available top-down. The declarative knowledge from ConceptNet and the structure of the graphical model are both qualitative and at a comparable level of abstraction, relative to the quantitative information extracted via the parameters. So, we have restricted ourselves to extracting only the structure in the form of conditional dependencies from data and ignore parameter estimation in the form of conditional probability values.

To conclude, in this Chapter, we were able to model traffic dynamics (variations) utilizing a variant of LDS called RSLDS. RSLDS approximates non-linear dynamics in traffic variations utilizing piecewise linear approximation based on the knowledge of the domain. We build normalcy models for traffic dynamics utilizing a massive real-world data and thereby tag anomalies. We demonstrated the role of events extracted from textual observations in interpreting anomalies in traffic dynamics. While modeling traffic dynamics, we introduced a latent variable to account for lack of knowledge of event interactions. In the later part of the Chapter, we explored event interactions thereby gaining insights into the composition of the latent variable.

6

Action Recommendation

"The key to good decision making is evaluating the available information - the data - and combining it with your own estimates of pluses and minuses. As an economist, I do this every day."

— Emily Oster

Action recommendation in Physical-Cyber-Social systems involve decision making utilizing observations of the user and the environment along with the domain knowledge. The role of knowledge is analogous to the pluses and minuses mentioned in the above quote by the American Economist, Emily Oster. PCS Systems typically have massive multimodal and heterogeneous observations spanning the physical, cyber, and social dimensions. The insights synthesized from these observations would be useful only when the consumer of these observations is able to perform an action to meet the goals. Ideally, these actions should adapt the dynamic nature of the PCS domain. For example, consider the problem of recommending optimal action to a user for a Do It Yourself (DIY) task, where the user wants to perform a task in an IoT (Internet of Things) environment. There are two important things to consider before recommending a task: i) Resources required to accomplish the task and their availability in the IoT environment accessible to the user. ii) Skills possessed by the user and their relevance to the task at hand. Action recommendation should cope with the

Figure 6.1: Action Recommendation for the task of making a French Toast

dynamism and uncertainties in IoT environments.

We will explore some of the ideas in the reinforcement learning [138] literature to solve the problem of optimal action recommendation in IoT environments.

6.1 Action Recommendation Engine

A task recommendation engine should be able to provide an optimal action to the user based on the available resources and known user skills. We need three important components to develop a task recommendation engine as shown in Figure 6.2: i) A language to represent tasks. ii) An algorithm to recommend optimal action. iii) An evaluation environment for task recommendation.

The task store in Figure 6.2 contains tasks represented in machine readable and interchangeable format. Semantic web technologies provide standards for knowledge representation such as Resource

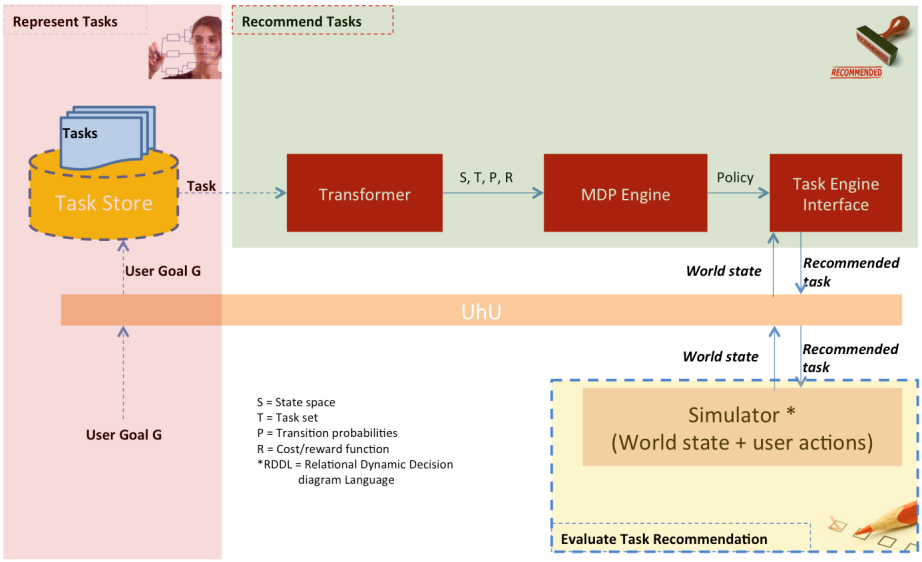


Figure 6.2: Architecture for the Action Recommendation Engine

Description Framework[1] (RDF), RDF Schema[2] (RDFS), and Web Ontology Language[3] (OWL). We utilize RDFS description for representing tasks. Given a user goal, G, task representation, which may contain a sequence of actions, is retrieved from the task store. Task representation is transformed into a formal model of Markov Decision Process (MDP) which is utilized to recommend the optimal action to the user. Evaluation of task recommendation requires tracking the evolution of the world state upon user actions subject to the stochasticity in the world states. We configure and utilize a stochastic simulation engine to facilitate an environment for evaluating task recommendation. We will describe each of the three components in detail in the following sections.

6.1.1 Task and Action

A task is a high level description of the state that the user wants to achieve in an IoT environment. For example, preparing French toast is a task. Actions are an ordered steps to be taken to perform

[1] https://www.w3.org/RDF/
[2] https://www.w3.org/TR/rdf-schema/
[3] https://www.w3.org/TR/owl-features/

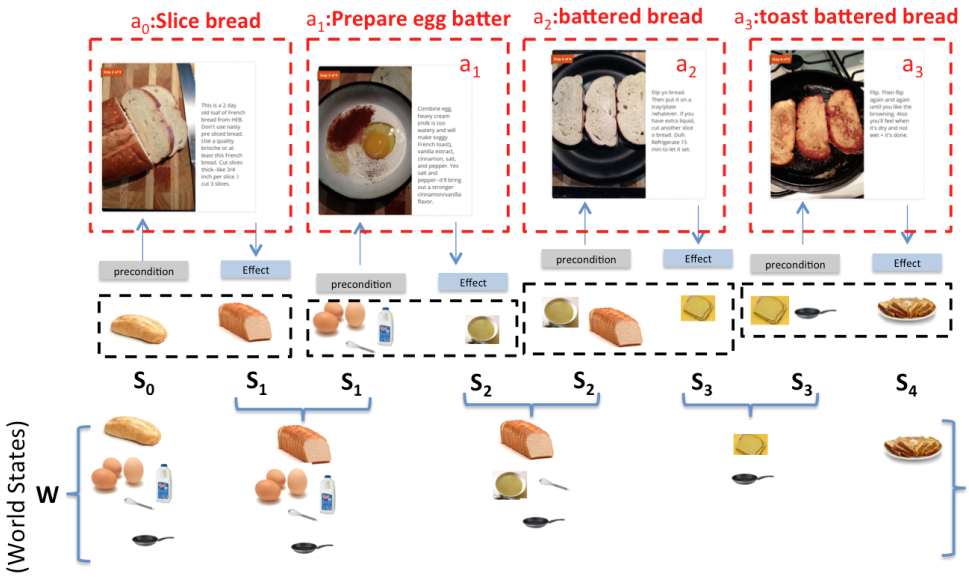


Figure 6.3: Pre- and post-condition based action representation for the task of making a French Toast representing slice bread, prepare egg batter, prepare battered bread, and toast the battered bread as tasks.

a task. For example, slice bread, prepare egg batter, prepare battered bread, and toast battered bread are the actions to be performed in the same order to complete the task of preparing French toast.

6.1.2 Optimal Action

We stated that the action recommendation is desired to be optimal to the users intending to perform a particular task in the IoTs environment. Action recommended to the user should be such that it matches the skill level of the user. If a user is a novice, the actions recommended should be simple enough for the user to follow. If a user is an expert, then the action recommended should match the appropriate granularity of details in performing the task. Action recommendation should also match the availability of resources in the IoT environment. An optimal action is the one that the user can perform with maximum success probability utilizing the available resources in the IoT environment.

6.2 Representation of Actions

Action recommendation for the task of making a French toast as given by Snapguide[4] is shown in Figure 6.1. These actions are static, unresponsive to changes in the environment and not personalized to the skills of the user performing the task. A better action representation should capture the domain knowledge and further, accommodate dynamism and uncertainty in the IoT environment. Further, such an action representation should be based on action completion. That is, the next action to be recommended to the user depends on the current action that is completed. Completion based action recommendation ignores the possibility of action reordering and action failure. Action completion based recommendation is not flexible (due to strict order) and responsive (due to task failure agnostic behavior) to events in the IoT environment.

Actions can be represented using pre- and post-conditions for each action [152; 6; 98]. A sequence of actions can then be performed based on the satisfiability of pre-conditions eventually leading to the completion of a task. User can potentially choose from a pool of actions satisfying the pre-conditions. Such a representation offers flexibility and can potentially perform responsive action recommendation. Pre- and post-condition based action recommendation is shown in Figure 6.3. Slice bread, prepare egg batter, prepare battered bread, and toast the battered bread are the four tasks represented using pre- and post-conditions. Pre-conditions specify the world state in which the action can be performed, e.g., denoting the required resources and constraints for performing the action. Post-condition specifies the effect of performing the action on the world state, e.g., prepare egg batter task when performed successfully, results in egg batter being available in the world state as the post-condition. Formally, world state may be represented as a set of state variables $\mathbf{W}$ as shown in Figure 6.3 used to track the evolution of the world state consisting of available resources in the environment. User actions along with the stochastic nature of the world affects the world state $\mathbf{W}$ and its evolution. User actions may add resources or modify/transform existing resources

[4] https://snapguide.com/

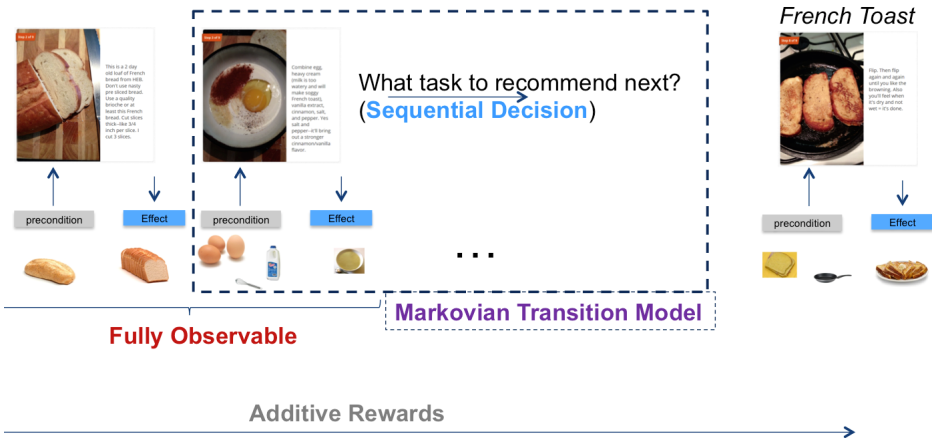


Figure 6.4: Nature of the action recommendation problem and its connection to Markov Decision Process (MDP).

reflected in **W**, e.g., prepare egg batter action results in addition of egg batter resource to the environment. Stochastic nature of the environment may modify the resources in the environment, e.g., milk can go bad over time and hence unavailable as a resource reflected in **W**.

A semantic representation of pre- and post-condition allows for applying inference rules in evaluation of pre- and post-conditions. For example, consider the prepare egg batter action in Figure 6.3 with whisk being absent. However, there is fork in the environment which is a type of egg beating device. A pre-condition evaluation without the knowledge that a fork is a type of egg beating device results in the failure of pre-condition. The task recommendation is interrupted even though the prepare egg batter action can still be performed. Semantics enabled pre-condition evaluation would reuse the existing knowledge and some commonsense knowledge to evaluate the pre-condition. In this scenario, absence of whisk and presence of fork in the environment still results in successful evaluation of the pre-condition resulting in uninterrupted action recommendation.

6.3 Finding Optimal Action

6.3.1 DIY Task Recommendation: Nature of the Domain

First, we will consider the nature of the problem we are trying to solve for action recommendation. A user is performing a task and we need to provide the optimal action to the user given the current world state to accomplish the task. In a realistic setting, a user may fail at performing a task (e.g., charred bread toast) resulting in stochastic transitions between various intermediate world states. The problem of optimal action recommendation is depicted in Figure 6.4. Action recommendation has to be performed at every step over a sequence of states. Hence, it is a sequential decision problem. The best possible action recommendation at the current state depends on the current world state alone and not on the way in which the current state is attained. Hence, it is a Markovian transition model. We assume that the environment is fully observable, i.e., the world state **W** is a faithful representation of the real-world. Further, as the user performs actions, the user is progressing toward task completion captured using the additive rewards. To summarize, we are looking at a sequential decision problem which is fully observable with Markovian transitions and additive rewards; a.k.a MDP.

6.3.2 Task Description to Actions

The action recommendation engine as shown in Figure 6.2 consists of the task description in the task store which we also refer to as the domain knowledge. Tasks are described in terms of the actions required to be performed to accomplish the task. The preconditions of actions provides information on the resources required in the environment which is part of the world state. Further, we may have information on the skills required by the user to complete an action. Resource requirements and user skills are transformed into MDP, a probabilistic graphical model. The transformation of the knowledge represented in the form of tasks and associated actions along with its pre- and post-conditions to an MDP problem is the key contribution of this work. The policy generated by the

Table 6.1: Relational Dynamic Influence Diagram Language (RDDL) description of making egg batter action along with the world states and its evolution.

```
// Objects
types environment: object; ;
// non-fluents
TASK-SUCCESS-PROB :  non-fluent, real, default = 0.5 ;
// States
resource_milk(environment) :  state-fluent, bool, default = false ;
resource_egg_batter(environment) :  state-fluent, bool, default = false ;
// transition functions
resource_milk'(?e) = KronDelta(resource_milk(?e) );
resource_egg_batter'(?e) =
if (resource_egg_batter(?e) ) then KronDelta ( true )
else if ( create_egg_batter(?e) )
KronDelta ( resource_milk(?e) ∧ resource_eggs(?e) ∧ Bernoulli(TASK-SUCCESS-PROB ) )
else KronDelta (false);
// reward function
reward = [sum_?e : environment  ( goal-state(?e) * PENALTY-NON-GOAL-STATE)];
```

MDP engine is used to recommend actions to the user. We utilize the formalisms of MDP described in Chapter 3 to find the best possible action at each state.

6.3.3 Parameterizing MDP

The transformer in Figure 6.2 takes task description as input and provides a parameterized MDP as output. An MDP consists of set of states (S), set of actions (A), transition probabilities (P), and reward function (R) represented using the tuple $\langle \mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R} \rangle$. **S** represents all possible values of the world state **W** including the initial state, intermediate states, and the final state. **A** represents all possible actions that can be performed at all possible states. **P** represents the transition probabilities between various states upon the performance of action at each state. **R** represents the reward function that captures rewards for attaining a particular state by performing a certain action.

We utilize the task description to initialize the parameters of the MDP: $\langle \mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R} \rangle$. **S** is all possible states, $\{S_0, S_1, S_2, S_3, S_4\}$ for the scenario in Figure 6.3. **A** represents the set of all possible actions, $\{a_0, a_1, a_2, a_3\}$ representing slice bread, prepare egg batter, prepare battered bread, and toast the battered bread, respectively. The transition probabilities, **P**, represent the probability of

moving from one state to the other when an action is performed. This information can be gleaned from observations or in our case, by initializing based on the difficulty of performing the action. For example, we can guess a relative order of the difficulty of action to be $\{a_0, a_2, a_3, a_1\}$. Slicing bread action a_0 is probably the simplest of all the tasks. Preparing battered bread action, a_2, comes next since it is not as simple as slicing bread may be because the uniformity needs to be maintained and so as not to get the bread soggy. The next level of difficulty may be the toast battered bread action, a_3, since it requires the knowledge of using the pan with heat and making sure that the toast is uniformly roasted on both sides. Preparing egg batter, a_1, is the hardest of all these actions since it involves breaking of eggs, mixing milk, and beating it steadily. This total ordering is crucial but the values we assign for the transition probabilities is immaterial. The reward function, $\mathbf{R}$, must indicate reward for all possible actions leading to various state transitions. The desired world state is given the highest reward and again, the relative ordering of rewards for actions are important rather than the exact reward values.

6.4 Evaluating Action Recommendation

Evaluating task recommendation is a challenging task due to the intricate interactions between user skills (gleaned from user interactions with the environment; this is out of scope of the work being discussed here), stochastic environment, and user actions. User skills play a crucial role in the type of actions users can be recommended to perform. For example, an expert user may need very coarse grained instructions while a novice may need very fine grained action recommendation. Success of user actions depends not just on the user skills but also on the stochasticity of the environment. For example, performing simple tasks may be challenging in dynamic environment. For a realistic evaluation of action recommendation we need to model the stochastic environment and the user actions and its effect on the environment.

Environment and effect of user actions on the environment is well represented and studied in

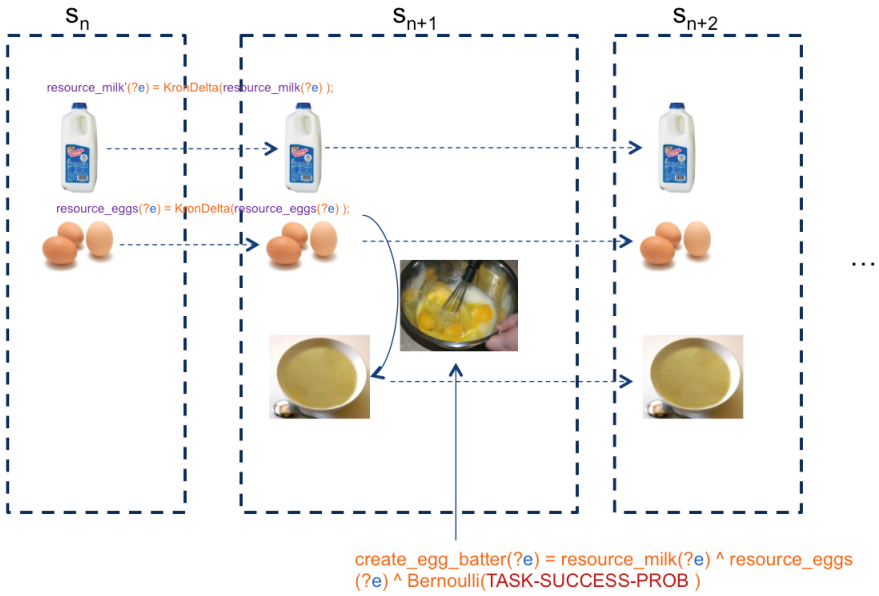


Figure 6.5: RDDL intuition in pictures.

the domain of AI planning. For example, Planning Domain Definition Language (PDDL) [101] is an attempt to normalize description of planning problems for a shared understanding. PDDL has enabled International Planning Competitions creating a unified competitive spirit for solving some challenging planning problems [53]. PDDL represents the environment as a collection of states, user actions, pre-condition for user actions, and post-conditions of user actions as state transitions along with many other functionalities. For stochastic domains (such as in our environment, exemplified below), PDDL is extended to PPDDL [153] to capture uncertainties in the planning environment. RDDL is motivated by the dynamic transition idea of Dynamic Bayesian Networks (DBNs) [41; 105].

A sample RDDL file consisting of state variables capturing resources in the environment and the effect of user action on state variables is shown in Table 6.1. The RDDL description looks convoluted at the first glimpse so, we will provide an intuitive description in terms of resources in the environment, user skills, and user actions as shown in Figure 6.5. Specifically, we consider the example of making French Toast. Milk and eggs in the environment can be represented using state

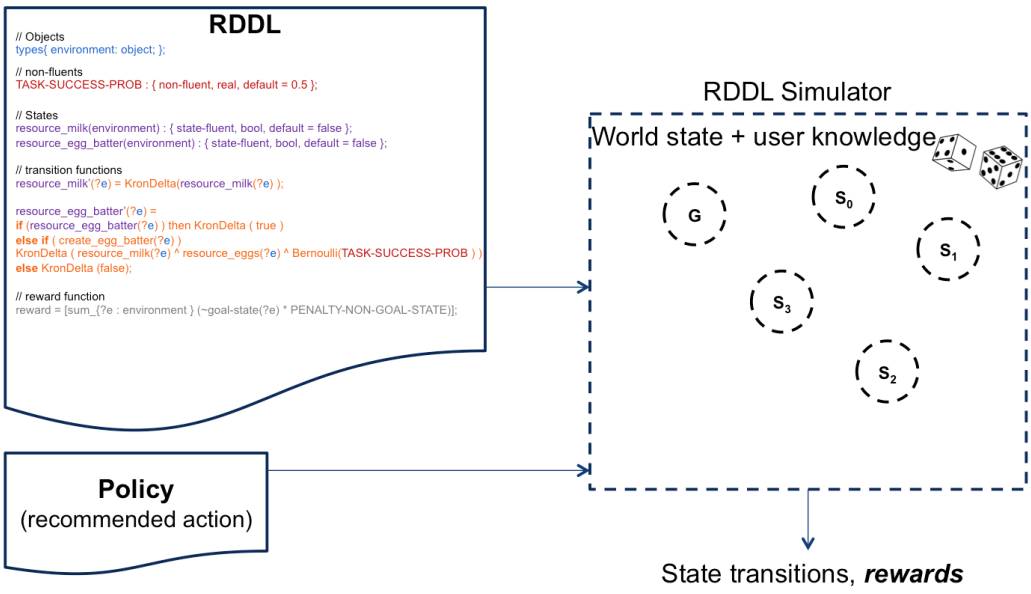


Figure 6.6: Working of a RDDL simulator used as part of the evaluation environment for action recommendation.

variables. Milk and eggs stays in the environment unchanged over time, unless the milk goes bad or someone drops the eggs. These are just two examples of stochasticity in the environment. This evolution is shown as state sequences $S_n, S_{n+1}, S_{n+2}, ..., S_{n+m}$ each containing state variables. In state S_{n+1}, the resources in the environment, eggs and milk is transformed into egg batter when the user performs the prepare egg batter action. The egg batter preparation success depends on the resource requirements and the task success probability defined in RDDL. Once the egg batter is prepared, it is added as a resource in the environment and it continues to stay in the environment till it is disrupted, as shown in Figure 6.5.

RDDL simulation engine[5] takes RDDL description of the environment, user skills, and user actions as input and provides a stochastic simulation of a realistic action recommendation environment. The simulation engine also needs a policy as input which will be utilized to perform appropriate actions depending on the wold state $\mathbf{W}$.

[5]https://github.com/ssanner/rddlsim

The ideas described in this Chapter can be extended to problems that can be modeled as reinforcement learning. Specifically, scenarios in which an agent has to act under uncertainty and the action results in some state which decides the reward. For example, in the context of asthma management, we can model the problem of minimizing asthma attacks in children using a MDP model. The most favorable state is the ones which the patient has no asthma attacks. The action in this scenario may include taking preventive medication, avoiding certain locations with high pollen content, avoiding exposure to poor air quality, vacuuming indoors, and using a humidifier.

7

Conclusion and Future Work

Understanding real-world event interactions and their dynamics from observational data in PCS systems is a challenging problem. Some of the challenges include *uncertainty*, *incompleteness*, *heterogeneity*, and *dynamism* of real-world events and their manifestations in observational data. We demonstrated that probabilistic graphical models are a natural fit to deal with these challenges in PCS systems. PGMs utilize probability as a calculus for dealing with *uncertainty* and graph structure to capture complex event interactions and event dynamics. We demonstrated the use of temporal probabilistic graphical models in modeling *dynamism* of PCS events. While utilizing PGMs for addressing these challenges, we discovered that the declarative domain knowledge can complement learning structure and parameters of PGMs. The *incompleteness* and *heterogeneity* challenges were dealt by integrating observations from people and machine sensors.

Our solution has three components: i) Event extraction, ii) Event understanding, and iii) Action recommendation for processing observational data from PCS systems. As mentioned earlier, these three solution components correspond to the John Boyd's OODA-loop operations of observe, orient, and decide & act respectively. The three solution components are grounded in a variety of PGMs enumerated in Figure 7.1.

A high level view of various components used to analyze PCS systems is provided in Figure 7.1.

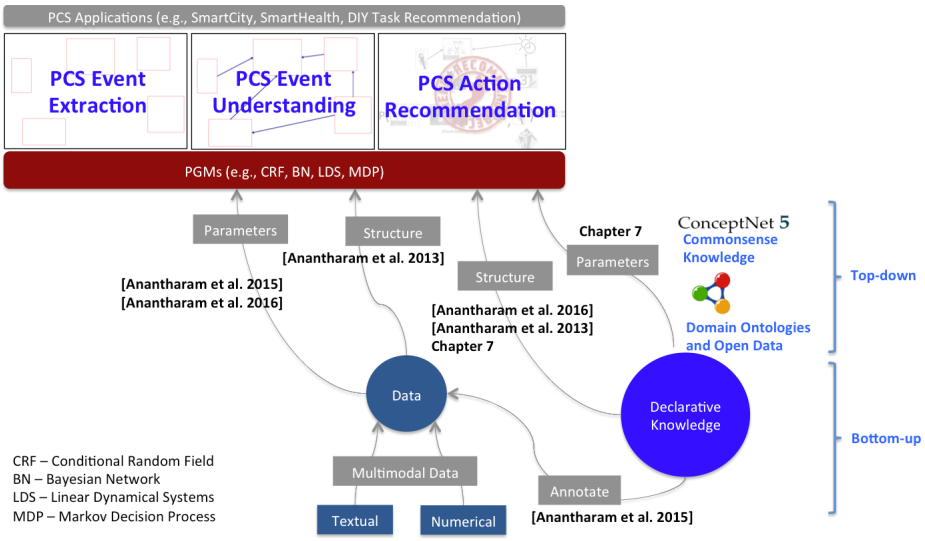


Figure 7.1: Relationship between PGMs, declarative knowledge, and PCS system along with the publications indicating the topic of contribution with respect to the overall thesis idea of supporting PGMs with declarative knowledge

Interactions between observational data, declarative knowledge, PGMs, three solution components, and PCS applications introduced earlier are presented here for clarity. As noted earlier, PGMs are specified using structure and parameters. Each of the publication mentioned in Figure 7.1 utilizes the indicated method of deriving structure and parameters. PCS event extraction work [8] discussed in Chapter 4 addresses annotation of raw data and learning parameters of a CRF model. The PCS event understanding work [10] discussed in Chapter 5 deals with learning parameters for an LDS model utilizing domain knowledge for specifying the structure. We presented declarative knowledge enabled BN structure extraction to uncover event interactions from data [11] in Chapter 5. Chapter 6 introduced the ideas for initializing parameters of a MDP model utilizing declarative knowledge. We demonstrated that both top-down and bottom-up techniques enable specification of PGMs, and specifically, declarative domain knowledge can support both structure and parameter specification of PGMs.

7.1 Conclusions

First, we conclude with various observations we assimilated during PCS event extraction, understanding, and action recommendation in this section.

7.1.1 PCS event extraction

Real-world events manifest in multiple modalities such as observations made from machine sensors and observations made by people. One modality may provide corroborative information with respect to the other modality. One modality may provide complementary information with respect to the other modality. More importantly, one modality can be utilized to facilitate interpretation of other modality, e.g., accident event extracted from observations from people can be utilized to interpret dip in average speed of vehicles passing through the accident prone link. We found that observations from people may provide qualitative information (e.g., hot) while observations from sensors provide quantitative observations (e.g., 100 degree Fahrenheit). We also observed that people report various infrastructure related issues in a city on social media. To tap into observations from people, we devised event extraction techniques over short text data stream such as tweets and SMS messages.

We formulated the problem of extracting events from tweets as a two step process of annotation and spatio-temporal aggregation. We utilized a sequence labeling technique, the Conditional Random Field (CRF) model, to identify location names and event terms of interest in tweets collected from a city. The parameter learning of CRF model, a type of probabilistic graphical model, was supported by declarative knowledge of locations and event terms of interest by creating large training datasets. Further, our aggregation algorithm was able to synthesize traffic events from annotated tweets. We showed that we can indeed extract city related events from tweets. Specifically, we evaluated our approach on city traffic related events due to the availability of open data and ground truth. Our evaluations focused on comparing events we extracted from tweets with the events reported by formal city sources. We found promising results with extracted events being

complementary, corroborative, and timely with respect to formal sources reporting on city events.

Availability of twitter data is not guaranteed in the developing world to glean people's reports of various city related events. However, SMS (Short Message Service) messages are widely used by people in developing part of the world. We observed that SMS messages can indeed be utilized to extract traffic related incidents in a city. Traffic incidents were extracted from near real-time SMS messages provided by city authorities. We were able to study the impact of these traffic related incidents on the schedule of public transport vehicles. Using this information we were able to provide event aware route recommendation to plan a journey in the city.

7.1.2 PCS event understanding

We realized the crucial role of observations by people in understanding real-world events in the event extraction step. For making sense of real-world events, we need to process both machine sensor data and observations from people in a unified setting. Toward this integration, we devised techniques to utilize observations from people to interpret variations in sensor observations. Specifically, we were able to interpret traffic dynamics (variations in average speed of vehicles) in terms of traffic events reported by people in a city. In fact, we found that coverage of people's report of incidents is much broader than official reports of incidents by city sources. Further, we found that the influence of long term events on traffic may not manifest in sensor observations depending on the duration. This is because influence of long term events may already be part of normalcy model generated over limited duration and hence requires a different source for detecting them from data. However, we were able to successfully detect manifestation of short term events using just the traffic sensor data.

Event interactions are crucial for understanding intricate nature of complex systems. We addressed the challenge of uncovering event interactions from observational data. Specifically, we demonstrated a scenario in understanding interactions between various traffic related events by formulating it as a BN structure extraction problem. We also addressed one of the limitations of purely data driven approach to uncover the true structure of interactions between city traffic related events

by proposing a declarative knowledge driven approach to complement BN structure extraction algorithms. This hybrid approach utilized three operations to disambiguate between various structures that are equally plausible as they have the same likelihood scores.

7.1.3 PCS action recommendation

Recommendation for Do-It-Yourself (DIY) tasks has received significant attention of late and we demonstrated its role in an Internet Of Things (IoT) environment in Chapter 6. Increasingly, resources are going to be distributed and available as and when needed. Task recommendation in such a distributed environment is challenging due to dynamism (changing available resources), uncertainty (varying expertise of a new user), incompleteness (task success or failure), and heterogeneity (multiple types of resources). To address these challenges, we proposed a task representation language that uses Semantic Web technologies, a task recommendation algorithm to recommend an optimal action to the user, and an environment to evaluate the task recommendations.

Semantic Web technologies provides a standardized language for representation, exchange, and inference. A distributed environment like IoT demands standardization of task representation thereby facilitating reuse of task descriptions. Further, Semantic Web languages provide reasoning capabilities that can be utilized for task recommendation. Finally, existing knowledge of various domains can be reused for task representation, e.g., resources in the environment can be mapped to resources on Freebase.

We formulated the problem of recommending optimal action as a Markov Decision Process (MDP) problem. We derived the parameters of the MDP model by utilizing declarative task representation in a Semantic Web language. We demonstrated the role of task complexity in specifying the reward (inversely, cost) and transition matrix of a MDP model. This demonstrated the utility of declarative knowledge in synthesizing a probabilistic model for decision making.

In order to facilitate comparison of various task recommendation strategies, we configured a stochastic environment that reflected a DIY task environment. The stochasticity was captured in

the transition matrix entries which specified the probability of completing a task accounting for failures. We demonstrated that a stochastic simulation engine can be adapted to evaluate DIY task recommendation.

7.2 Future Work

We organize future research directions into three subsections for clarity.

7.2.1 PCS event extraction

We proposed techniques to create huge training data with minimum manual effort by leveraging declarative domain knowledge. We utilized the training data to train the CRF model to identify locations and event terms. We focused on extracting city traffic related events for evaluating our event extraction algorithms due to the availability of ground truth data. One future direction is to extract all city related events concerning entertainment, infrastructure, weather, crime, water, and power grid. The extracted events in each category can be compared with ground truth events from open city data to evaluate the success of event extraction. These events can be utilized for better situational awareness by city authorities and people.

Deep Learning has performed better than various state-of-the-art approaches in tasks such as image processing, speech recognition, and game playing. Large quantity of data is paramount in deep learning and unavailability of data can be a significant hindrance. Given the ability to create massive training data automatically using our approach, one possible exploration direction is to utilize deep learning approach for sequence labeling [145; 136] tasks. It would be interesting to compare deep learning based annotation model with the CRF annotation model.

While social media is a great source of real-world events as demonstrated in this dissertation, there are several challenges in utilizing them. Some of the challenges include data quality, trustworthiness, and redundant and biased event reports. It would be interesting to explore data quality issues while

utilizing observations from people. One possible research direction is to explore the trustworthiness issues associated with various real-world events. Trustworthiness of the reported events is crucial for decision making in crisis or in the presence of an adversary. Some events may get reported more often compared to other events resulting in propagation of "popular" events. Understanding such biases would provide ways to calibrate how we synthesize information from reported events.

7.2.2 PCS event understanding

We presented techniques to correlate anomalies in sensor data with events extracted from textual observations from a city. Understanding the dynamics of each type of event for later detection from sensor data alone is an interesting research direction. For this, the tagged anomalies in sensor data should be studied in detail to identify patterns or trends. Further, the idea of interpreting variations in quantitative data (observations from sensors) utilizing qualitative data (observations from people) can be explored across various application domains such as healthcare and system health monitoring.

The anomaly detection models are essentially learned from data and do not rely on any manual specification of anomalies. This implies that the data we utilize to create the anomaly detection models dictates the criteria for tagging anomalies. The validity of the anomaly detection model has to be explored based on the contextual relevance. For example, creating models specific to each season such as summer vs. winter as opposed to having a single model for the entire year would be an interesting idea to explore. Introduction of such context specific models of different granularity can enhance model stability and relevance. Further, the anomaly detection model needs to be updated (e.g., to accommodate completion of long term event) and the criteria leading to this update requires further consideration.

The LDS model for capturing traffic dynamics was proposed in a systematic way by examining the theoretical nature of the problem. In the process of modeling traffic dynamics, we utilized one hidden variable (volume of vehicles passing through a link) and one observed variable (average speed of vehicles passing through a link). However, the real-world is much more complex with

lot of interacting events as explored in Chapter 5. Examining traffic dynamics by utilizing the event interaction structure extracted from data can provide interesting insights. Each time slice in LDS had a hidden and an observed variable. Event interactions extracted from data can provide additional insights into the hidden variable revealing intricate event interactions.

7.2.3 PCS action recommendation

We explored the domain of task recommendation for DIY tasks in the context of IoT environment. The idea of initializing sequential decision making problem like MDP utilizing declarative knowledge of tasks can be explored in other domains. For example, MDP can be utilized to model the problem of minimizing asthma attacks in a patient. Actions may include taking medication, staying indoor, using humidifier, and changing route of commute to work. The reward function in this scenario will be based on the occurrence of asthma attacks. However, there are many challenges in defining the state space and set of all possible actions in this scenario. One possible research direction can be on investigating the utility of declarative knowledge for specifying the asthma problem using MDP formalism.

Reinforcement learning has received significant attention due to AlphaGo [131], the first system to defeat a human world champion[1] of the ancient Chinese game, Go. A combination of Deep Learning along with reinforcement learning techniques were utilized to create AlphaGo by Google Deep Mind team. One possible research direction is to investigate the problem of asthma management along the lines of game playing. In the asthma scenario, the environment plays the role of the opponent player in a game. With enough data from environment and information of actions and asthma attacks for a person, an optimal action recommendation can be learned using reinforcement learning techniques.

We found that real-world events are reported across various observational modalities. For a better understanding and situational awareness, we need to process multiple observational modalities. Throughout the dissertation, we proposed the idea of extracting events, understanding event inter-

[1] http://www.theatlantic.com/technology/archive/2016/03/the-invisible-opponent/475611/

actions and dynamics, and formalized the problem of action recommendation in PCS systems. All the ideas were grounded in the form of a probabilistic graphical model specified utilizing declarative knowledge or learned from observational data. The abstract ideas and the concrete algorithms presented in this dissertation benefits problems in domains such as traffic analytics, power grid management, healthcare, and system health monitoring.

References

511.org. San Francisco 511 service, 2015. Available online: `http://511.org` (Accessed Nov 4, 2015).

Charu C Aggarwal and Karthik Subbian. Event detection in social streams. In *SDM*, volume 12, pages 624–635. SIAM, 2012.

Alfred V Aho and Margaret J Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, 1975.

E.M. Airoldi. Getting started in probabilistic graphical models. *PLoS Computational Biology*, 3(12):e252, 2007.

Alias-i. Lingpipe 4.1.0, 2008.

Jose A Ambros-Ingerson and Sam Steel. Integrating planning, execution and monitoring. In *AAAI*, volume 88, pages 21–26, 1988.

Pramod Anantharam, Tanvi Banerjee, Amit Sheth, Krishnaprasad Thirunarayan, Surendra Marupudi, Vaikunth Sridharan, and Shalini G Forbis. Knowledge-driven personalized contextual mhealth service for asthma management in children. In *IEEE 4th International Conference on Mobile Services*, 2015.

Pramod Anantharam, Payam Barnaghi, Krishnaprasad Thirunarayan, and Amit Sheth. Extracting city traffic events from social streams. *ACM Trans. Intell. Syst. Technol.*, 6(4):43:1–43:27, July 2015.

Pramod Anantharam and Biplav Srivastava. City notifications as a data source for traffic management. In *Proceedings of the 20th ITS World Congress 2013*. 2013.

Pramod Anantharam, Krishnaprasad Thirunarayan, Surendra Marupudi, Amit Sheth, and Tanvi Banerjee. Understanding city traffic dynamics utilizing sensor and textual observations. *In Proceedings of The Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), February 12–17, Phoenix, Arizona, USA*, 2016.

Pramod Anantharam, Krishnaprasad Thirunarayan, and Amit P Sheth. Traffic analytics using probabilistic graphical models enhanced with knowledge bases. *In Proceedings of 2nd International Workshop on Analytics for Cyber-Physical Systems (ACS-2013) at SIAM International Conference on Data Mining (SDM13), May 2-4, Texas, USA*, 2013.

Jessica Anderson and Michael Bell. Travel time estimation in urban road networks. In *Intelligent Transportation System, 1997. ITSC'97., IEEE Conference on*, pages 924–929. IEEE, 1997.

Fahiem Bacchus and Adam Grove. Graphical models for preference and utility. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 3–10. Morgan Kaufmann Publishers Inc., 1995.

Daniel Balageas, Claus-Peter Fritzen, and Alfredo Güemes. *Structural health monitoring*, volume 493. Wiley Online Library, 2006.

David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

Matthew Barth and Kanok Boriboonsomsin. Algorithm for finding optimal paths in a public transit network with real-time data. In *Jour. TRB, No. 2256, TRB*, 2011.

BBC News. Missing Malaysia Airlines plane 'a mystery', 2014. Available online: `http://www.bbc.com/news/world-asia-26510027` (Accessed Nov 4, 2015).

Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *ICWSM*, 2011.

Jennifer Bélissent. ,Äúgetting clever about smart cities: new opportunities require new business models. 2010.

Bélissent, Jennifer. Service Providers Accelerate Smart City Projects, 2013. Available online: `http://bit.ly/1NPvPry` (Accessed May 26, 2016).

Richard Bellman. A markovian decision process. Technical report, DTIC Document, 1957.

J. Bilmes and G. Zweig. The graphical models toolkit: An open source software system for speech and time-series processing. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 4, pages IV–3916– IV–3919. IEEE, 2002.

Susanne Biundo, Pascal Bercher, Thomas Geier, Felix Müller, and Bernd Schattenberg. Advanced user assistance based on ai planning. *Cognitive Systems Research*, 12(3):219–236, 2011.

C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee. Linked data on the web (ldow2008). In *Proceeding of the 17th international conference on World Wide Web*, pages 1265–1266. ACM, 2008.

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *International journal on semantic web and information systems*, 5(3):1–22, 2009.

Jim Blythe. An overview of planning under uncertainty. In *Artificial intelligence today*, pages 85–110. Springer, 1999.

Jennifer Boger, Pascal Poupart, Jesse Hoey, Craig Boutilier, Geoff Fernie, and Alex Mihailidis. A decision-theoretic approach to task assistance for persons with dementia. In *IJCAI*, pages 1293–1299. Citeseer, 2005.

Craig Boutilier, Thomas Dean, and Steve Hanks. Planning under uncertainty: Structural assumptions and computational leverage. In *Proceedings of the Second European Workshop on Planning*, pages 157–171. Citeseer, 1995.

John R Boyd. The essence of winning and losing. *Unpublished lecture notes*, 1996.

Jeffrey A Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B Srivastava. Participatory sensing. 2006.

Brian Caulfield and Margaret Mary O'Mahony. Factors influencing preferences for real-time public transport information. In *European Transport Conference*, pages 1–11, 2007.

Sanjay Chawla, Yu Zheng, and Jiafeng Hu. Inferring the root cause in road traffic anomalies. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 141–150. IEEE, 2012.

Edwin Chen. Introduction to conditional random fields, 2012.

Mei Chen, Xiaobo Liu, and Jingxin Xia. Dynamic prediction method with schedule recovery impact for bus arrival time. In *Transportation Research Record: Jour. TRB*, 2005.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968.

Diane J Cook and Sajal K Das. How smart are our environments? an updated look at the state of the art. *Pervasive and mobile computing*, 3(2):53–73, 2007.

Elizabeth M Daly, Freddy Lecue, and Veli Bicer. Westland row why so slow?: fusing social media and linked data sources for understanding real-time traffic conditions. In *Proceedings of the 2013 international conference on Intelligent user interfaces*, pages 203–212. ACM, 2013.

Brooks David. What data can't do, available online at: http://nyti.ms/yxln9d, 2013.

Brooks David. What you'll do next, available online at: http://nyti.ms/16znjy7, 2013.

Corrado De Fabritiis, Roberto Ragona, and Gaetano Valenti. Traffic estimation and prediction based on real time floating car data. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 197–203. IEEE, 2008.

Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5(2):142–150, 1989.

Wenwen Dou, K Wang, William Ribarsky, and Michelle Zhou. Event detection in social media data. In *IEEE VisWeek Workshop on Interactive Visual Text Analytics-Task Driven Analytics of Social Media Content*, pages 971–980, 2012.

DTP. Delhi traffic police updates. In *http://gupshup.me/groups/DTP?page=39*, 2012.

Charles Elkan. Log-linear models and conditional random fields, video lectures link:, 2008.

Kutluhan Erol, James Hendler, and Dana S Nau. Htn planning: Complexity and expressivity. In *AAAI*, volume 94, pages 1123–1128, 1994.

Leonhard Euler. *The seven bridges of Königsberg*. Wm. Benton, 1956.

Dave Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1:14, 2011.

Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2-3):113–127, 2014.

Eugene A Feinberg and Adam Shwartz. *Handbook of Markov decision processes: methods and applications*, volume 40. Springer Science & Business Media, 2012.

S. Fenz, A.M. Tjoa, and M. Hudec. Ontology-based generation of bayesian networks. In *Complex, Intelligent and Software Intensive Systems, 2009. CISIS'09. International Conference on*, pages 712–717. IEEE, 2009.

Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3):189–208, 1972.

Luca Filipponi, Andrea Vitaletti, Giada Landi, Vincenzo Memeo, Giorgio Laura, and Paolo Pucci. Smart city: An event driven architecture for monitoring public spaces with heterogeneous sensors. In *Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on*, pages 281–286. IEEE, 2010.

Alfonso E Gerevini, Patrik Haslum, Derek Long, Alessandro Saetti, and Yannis Dimopoulos. Deterministic planning in the fifth international planning competition: Pddl3 and experimental evaluation of the planners. *Artificial Intelligence*, 173(5):619–668, 2009.

Zoubin Ghahramani and Geoffrey E Hinton. Parameter estimation for linear dynamical systems. Technical report, Technical Report CRG-TR-96-2, University of Totronto, Dept. of Computer Science, 1996.

Google. General transit feed specification. at http://code.google.com/transit/spec/ transit_feed_specification.html, 2012.

Ralph Grishman, Silja Huttunen, and Roman Yangarber. Real-time event extraction for infectious disease outbreaks. In *Proceedings of the second international conference on Human Language Technology Research*, pages 366–369. Morgan Kaufmann Publishers Inc., 2002.

Raj Gupta, Biplav Srivastava, and Srikanth Tamilselvam. Making public transportation schedule information consumable for improved decision making. In *Proc. IEEE ITSC*, 2012.

Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18, 2008.

E.M. Helsper and L.C. van der Gaag. Building bayesian networks through ontologies. In *ECAI*, volume 2002, page 15th, 2002.

A. Hofleitner, R. Herring, P. Abbeel, and A. Bayen. Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network. *ITS, IEEE Transactions on*, 13(4):1679 –1693, dec. 2012.

E.J. Horvitz, J. Apacible, R. Sarin, and L. Liao. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. *arXiv preprint arXiv:1207.1352*, 2012.

Eric Horvitz, Johnson Apacible, Raman Sarin, and Lin Liao. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. In *In Twenty-First Conference on Uncertainty in Artificial Intelligence*, 2005.

Bingyan Huang, Tingting Zhao, and Yi Zhang. Traffic incident impact analysis with random matrix theory and cluster analysis. In *IEEE Conf. EMMS*, 2010.

R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of computational and graphical statistics*, 5(3):299–314, 1996.

M.B. Ishak, P. Leray, N.B. Amor, et al. Ontology-based generation of object oriented bayesian networks. In *Proceedings of the 8th Bayesian Modelling Applications Workshop*, pages 9–17, 2011.

Mouna Ben Ishak, Philippe Leray, and Nahla Ben Amor. A two-way approach for probabilistic graphical models structure learning and ontology enrichment. In *KEOD'11*, pages 189–194, 2011.

Amitabh Dixit Vishal Gupta James Macaulay Joseph Bradley, Christopher Reberger. Internet of everything (ioe): Top 10 insights from cisco,Äôs ioe value at stake analysis for the public sector. *CISCO Report*, 2013.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

Rudolf Emil Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics, Series A: Control*, 1(2):152–192, 1963.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Maurits Clemens Kaptein, Panos Markopoulos, Boris de Ruyter, and Emile Aarts. Persuasion in ambient intelligence. *Journal of Ambient Intelligence and Humanized Computing*, 1(1):43–56, 2010.

Michael Kehoe, Michael Cosgrove, SD Gennaro, Colin Harrison, Wim Harthoorn, John Hogan, John Meegan, Pam Nesbitt, and Christina Peters. Smarter cities series: a foundation for understanding ibm smarter cities. *An IBM Redguide publication*, 2011.

Predrag Klasnja and Wanda Pratt. Healthcare in the pocket: Mapping the space of mobile-phone health interventions. *Journal of biomedical informatics*, 45(1):184–198, 2012.

Joonho Ko and Randall L Guensler. Characterization of congestion based on speed distribution: a statistical approach using gaussian mixture model. In *Transportation Research Board Annual Meeting*. Citeseer, 2005.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 302–313. Morgan Kaufmann Publishers Inc., 1997.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Andreĭ Nikolaevich Kolmogorov. Foundations of the theory of probability. 1950.

Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 297–304. ACM, 2004.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

Vasileios Lampos and Nello Cristianini. Nowcasting events from the social web with statistical learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4):72, 2012.

H. Langseth and T.D. Nielsen. Fusion of domain knowledge with data for structural learning in object oriented domains. *The Journal of Machine Learning Research*, 4:339–368, 2003.

Freddy Lécué, Anika Schumann, and Marco Luca Sbodio. Applying semantic web technologies for diagnosing road traffic congestions. In *The Semantic Web–ISWC 2012*, pages 114–130. Springer, 2012.

Sangsoo Lee and Daniel Fambro. Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. *Transportation Research Record: Journal of the Transportation Research Board*, (1678):179–188, 1999.

Wei-Hsun Lee, Shian-Shyong Tseng, and Sheng-Han Tsai. A knowledge based real-time travel time prediction system for urban network. *Expert Systems with Applications*, 36(3):4239–4247, 2009.

W. Liao and Q. Ji. Learning bayesian network parameters under incomplete data with domain knowledge. *Pattern Recognition*, 42(11):3046–3056, 2009.

Greg Lindsay. Cisco's big bet on new songdo: creating cities from scratch. *Fast Company*, 1, 2010.

Hugo Liu and Push Singh. Conceptnet,Äîa practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

Mingrong Liu, Yicen Liu, Liang Xiang, Xing Chen, and Qing Yang. Extracting key entities and significant events from online daily news. In *Intelligent Data Engineering and Automated Learning–IDEAL 2008*, pages 201–209. Springer, 2008.

Seng W Loke. Building taskable spaces over ubiquitous services. *IEEE Pervasive Computing*, (4):72–78, 2009.

Jiakang Lu, Tamim Sookoor, Vijay Srinivasan, Ge Gao, Brian Holben, John Stankovic, Eric Field, and Kamin Whitehouse. The smart thermostat: using occupancy sensors to save energy in homes. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 211–224. ACM, 2010.

D. Madigan, J. Gavrin, and A.E. Raftery. Eliciting prior information to enhance the predictive performance of bayesian graphical models. *Communications in Statistics-Theory and Methods*, 24(9):2271–2292, 1995.

D. Margaritis. *Learning Bayesian network model structure from data*. PhD thesis, University of Pittsburgh, 2003.

A.A. Margolin, I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R.D. Favera, and A. Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC bioinformatics*, 7(Suppl 1):S7, 2006.

Lluís Màrquez, Xavier Carreras, Kenneth C Litkowski, and Suzanne Stevenson. Semantic role labeling: an introduction to the special issue. *Computational linguistics*, 34(2):145–159, 2008.

David Martin, Mark Burstein, Jerry Hobbs, Ora Lassila, Drew McDermott, Sheila McIlraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, et al. Owl-s: Semantic markup for web services. *W3C member submission*, 22:2007–04, 2004.

Ryusuke Masuoka, Bijan Parsia, and Yannis Labrou. Task computing–the semantic web meets pervasive computing. In *The Semantic Web-ISWC 2003*, pages 866–881. Springer, 2003.

Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. 2002.

Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. Pddl-the planning domain definition language. 1998.

Sharon Bertsch McGrayne. *The theory that would not die: how Bayes' rule cracked the enigma code, hunted down Russian submarines, & emerged triumphant from two centuries of controversy*. Yale University Press, 2011.

M. Miller and C. Gupta. Mining traffic incidents to forecast impact. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing*, pages 33–40. ACM, 2012.

CK Moorthy and BG Ratcliffe. Short term traffic forecasting using time series methods. *Transportation planning and technology*, 12(1):45–56, 1988.

Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

Meenakshi Nagarajan, Karthik Gomadam, Amit P Sheth, Ajith Ranabahu, Raghava Mutharaju, and Ashutosh Jadhav. Spatio-temporal-thematic analysis of citizen sensor data: Challenges and experiences. In *Web Information Systems Engineering-WISE 2009*, pages 539–553. Springer, 2009.

Milind Naphade, Guruduth Banavar, Colin Harrison, Jurij Paraszczak, and Robert Morris. Smarter cities and their innovation challenges. *Computer*, 44(6):32–39, 2011.

Dana S Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J William Murdock, Dan Wu, and Fusun Yaman. Shop2: An htn planning system. *J. Artif. Intell. Res.(JAIR)*, 20:379–404, 2003.

Martina Naughton, Nicholas Kushmerick, and Joseph Carthy. Event extraction from heterogeneous news sources. In *Proceedings of the AAAI Workshop Event Extraction and Synthesis*, pages 1–6, 2006.

Masayuki Okamoto and Masaaki Kikuchi. Discovering volatile events in your neighborhood: Local-area topic extraction from blog entries. In *Information Retrieval Technology*, pages 181–192. Springer, 2009.

N. Oliver and A.P. Pentland. Graphical models for driver behavior recognition in a smartcar. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 7–12. IEEE, 2000.

Øystein Ore. *Cardano: The Gambling Scholar*. Dover, 1965.

OSM. Open street map. In *In www.openstreetmap.org, en.wikipedia.org/wiki/OpenStreetMap, 2012.*

W Pattara-Atikom, P Pongpaibool, and S Thajchayapong. Estimating road traffic congestion using vehicle velocity. In *ITS Telecommunications Proceedings, 2006 6th International Conference on*, pages 1001–1004. IEEE, 2006.

Judea Pearl and T.S. Verma. A theory of inferred causation, 1991.

Vahe Poladian, Joao Pedro Sousa, David Garlan, and Mary Shaw. Dynamic configuration of resource-aware services. In *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on*, pages 604–613. IEEE, 2004.

Martha E Pollack. Intelligent technology for an aging population: The use of ai to assist elders with cognitive impairment. *AI magazine*, 26(2):9, 2005.

Pramod Anantharam. Extracting City Traffic Events from Social Streams, 2014. Available online: `https://osf.io/b4q2t/wiki/home/` (Accessed May 27, 2015).

John Pucher, Nisha Korattyswaroopam, and Neenu Ittyerah. The crisis of public transport in india: overwhelming needs but limited resources. *Journal of Public Transportation*, 7:95–113, 2004.

John A Quinn, Christopher KI Williams, and Neil McIntosh. Factorial switching linear dynamical systems applied to physiological condition monitoring. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(9):1537–1551, 2009.

Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999.

Alan Ritter, Oren Etzioni, Sam Clark, et al. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1104–1112. ACM, 2012.

C. Rotsos, J. Van Gael, A.W. Moore, and Z. Ghahramani. Probabilistic graphical models for semi-supervised traffic classification. In *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, pages 752–757. ACM, 2010.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

Earl D Sacerdoti. The nonlinear nature of plans. Technical report, DTIC Document, 1975.

Hassan Sayyadi, Matthew Hurst, and Alexey Maykov. Event detection and tracking in social streams. In *ICWSM*, 2009.

M. Scutari. Learning bayesian networks with the bnlearn r package. *Journal of Statistical Software*, pages 1–22, 2009.

Lui Sha, Sathish Gopalakrishnan, Xue Liu, and Qixin Wang. Cyber-physical systems: A new frontier. In *Machine Learning in Cyber Trust*, pages 3–13. Springer, 2009.

Amit Sheth. Citizen sensing, social signals, and enriching human experience. *Internet Computing, IEEE*, 13(4):87–92, 2009.

Amit Sheth, Pramod Anantharam, and Cory Henson. Physical-cyber-social computing: An early 21st century approach. *Intelligent Systems, IEEE*, 28(1):78–82, 2013.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Joao Pedro Sousa, Vahe Poladian, David Garlan, Bradley Schmerl, and Mary Shaw. Task-based adaptation for ubiquitous computing. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 36(3):328–340, 2006.

David J. Spiegelhalter. Bayesian graphical modelling: a case-study in monitoring health outcomes. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(1):115–133, 1998.

A.N. Srivastava and J. Han. *Machine learning and knowledge discovery for engineering systems health management*, volume 22. Chapman & Hall, 2011.

Shiliang Sun, Changshui Zhang, and Guoqiang Yu. A bayesian network approach to traffic flow forecasting. *Intelligent Transportation Systems, IEEE Transactions on*, 7(1):124–132, 2006.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Machine Learning*, 4(4):267–373, 2011.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Hristo Tanev, Jakub Piskorski, and Martin Atkinson. Real-time news event extraction for global crisis monitoring. In *Natural Language and Information Systems*, pages 207–218. Springer, 2008.

Austin Tate. Generating project networks. In *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 2*, pages 888–893. Morgan Kaufmann Publishers Inc., 1977.

Devendra Kumar Tolani, Murat Yasar, Asok Ray, and Vigor Yang. Anomaly detection in aircraft gas turbine engines. *Journal of Aerospace Computing, Information, and Communication*, 3(2):44–51, 2006.

I. Tsamardinos, C.F. Aliferis, A. Statnikov, and E. Statnikov. Algorithms for large scale markov blanket discovery. In *Proceedings of the sixteenth international Florida artificial intelligence research society conference*, pages 376–381, 2003.

I. Tsamardinos, L.E. Brown, and C.F. Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.

U.S.-Canada Power System Outage Task Force. Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations. Technical report, 2004. Available online: `http://1.usa.gov/1xVLXKA` (Accessed Oct 29, 2015).

Mengqiu Wang and Christopher D Manning. Effect of non-linear deep architecture in sequence labeling. In *IJCNLP*, pages 1285–1291, 2013.

Xiaofeng Wang, Matthew S Gerber, and Donald E Brown. Automatic crime prediction using events extracted from twitter posts. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 231–238. Springer, 2012.

Zhenyu Wang and David Garlan. Task-driven computing. Technical report, DTIC Document, 2000.

Mark Weiser. The computer for the 21st century. *Scientific american*, 265(3):94–104, 1991.

Wikipedia. Aeroperú Flight 603, 1996. Available online: `https://en.wikipedia.org/wiki/Aeroper%C3%BA_Flight_603` (Accessed Nov 4, 2015).

David E Wilkins. *Practical planning: extending the classical AI planning paradigm*. Morgan Kaufmann, 2014.

Sandeep Yaramakala and Dimitris Margaritis. Speculative markov blanket discovery for optimal feature selection. In *ICDM*, pages 809–812. IEEE, 2005.

John Yen and Jonathan Lee. A task-based methodology for specifying expert systems. *IEEE Expert*, 8(1):8–15, 1993.

Håkan LS Younes and Michael L Littman. Ppddl1. 0: An extension to pddl for expressing planning domains with probabilistic effects. In *In Proceedings of the 14th International Conference on Automated Planning and Scheduling*, 2004.

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association.

Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: concepts, methodologies, and applications. *ACM Transaction on Intelligent Systems and Technology (ACM TIST)*, 2014.